



**Application Program**

Y20-0093-1

**1130 Statistical System (1130-CA-06X)  
System Manual**

This manual provides detailed information on the logic used in each program of the 1130 Statistical System.

**RESTRICTED DISTRIBUTION**

## CONTENTS

1.0	Introduction	1
2.0	General System Flowchart Narratives	4
	A. Regression and Factor Analysis	4
	B. Analysis of Variance	5
	C. Orthogonal Polynomials	6
3.0	Detailed Flowchart Narratives	7
	A. Regression Analysis	7
	B. Factor Analysis	11
	C. Routines Used by Regression and Factor Analysis	24
	D. Analysis of Variance	28
	E. Orthogonal Polynomials	34
	F. Routines Used by All System Programs	40
4.0	Programming Notes	44
5.0	List of Switches	46
6.0	Program Listings	47
7.0	Flowcharts	86

### Second Edition

Y20-0093-1 is a minor revision incorporating Technical Newsletter Y20-0144 and does not obsolete Y20-0093-0 with Y20-0144.

Significant changes or additions to the specifications contained in this publication will be reported in subsequent revisions or Technical Newsletters.

**RESTRICTED DISTRIBUTION:** This publication, and the program to which it applies, are provided to IBM customers to meet their equipment capabilities and application needs. Distribution is limited to such customers and requires the approval of local IBM management.

Address comments concerning the contents of this publication to  
IBM, Technical Publications Department, 112 East Post Road, White Plains, N. Y. 10601

## 1.0 INTRODUCTION

Most subroutines in the 1130 Statistical System are documented by means of a flowchart. The exceptions are those very short routines, FORTRAN-coded, which can be easily understood from the listings, and the Assembly Language subroutines as noted in the index on the next page. The comments and flowcharts associated with the Assembly Language subroutines will be supplied upon request.

Figure 1 illustrates the various blocks used in the flowcharts and their particular meaning. Lines connecting these blocks are made up of periods. Arrows showing the direction of flow are represented by an X.

Connector symbols use the following conventions: four-digit symbols refer to chart symbol (two digits) and block (two digits). For example, ABH1 refers to block H1 on Chart AB. Two-digit symbols refer to a block on the chart where the reference appears. For example, H1 appearing on Chart AB refers to block H1 on the chart.

# Index of Subroutines

<u>NAME</u>	<u>LABEL</u> cc. 73-76	<u>FLOW CHARTS</u> Chart Symbol	<u>NARRATIVES</u> Page	<u>LISTINGS</u> Page
ANOV2	NOV2	EA	6, 30	68
ANOVA	NOVA	DW	5, 29	66
COREL	CORL	DB	4, 24	55
COVEC	CVEC	EN	17	87
DATRD	DTRD	**	41	49
FCTR	FCTR	DM, DN	5, 12	71
FCTR1	FCT1	EG	5, 13	73
FCTR2	FCT2	EL, EM	5, 15	76
FCTR3	FCT3	EP	5, 18	82
FMAT	FMAT	HH	42	85
FMTRD	FMRD	**	40	47
GDIV	GDIV	*	42	53
GET	GETO	DY	29	68
GMPY	GMPY	*	42	52
INVS	INVS	EH	13	73
MATIN	MATN	EZ, FA	22	83
MNSQ	MNSQ	EC, ED, EE	31	69
MXRAD	MXRD	DA	26	54
PCOEF	PCOF	DJ	37	57
PDER	PDER	DL	38	61
PFIT	PFIT	DK	39	60
POL2	POL2	DF	6, 36	58
POLSQ	PLSQ	DH	36	59
POLY	POLY	DD	6, 35	57
PROMX	PRMX	ES	19	78
PRNT	PRNT	DC	25	56
PRNTB	PRNB	**	41	49
QR	QR00	EJ	15	75
VARMX	VRMX	ET, EW, EX	18	79
VECTR	VCTR	EN	16	80
REGR	REGR	DM, DN	4, 8	62
REGR2	RGR2	DP	4, 9	63
REGRE	RGRE	DR, DS, DT	10	64
REPRT	RPRT	EF	32	70
RFOUT	ROUT	ER	20	77
RPRNT	RPNT	EY	21	82
SCORE	SCOR	FB	22	84
SDOP	SDOP	EB	30	69
STORE	STOR	DY	29	67
TRAN	TRAN	*	43	54
TRIDI	TRID	EJ	14	74
XMAX	XMAX	*	23	74

\* Item is not included; listing is considered to be of sufficient aid.

\*\* Item is not included; also, listings are not commented.



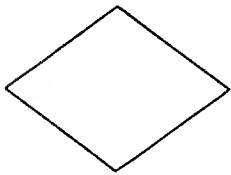
Enter or exit block



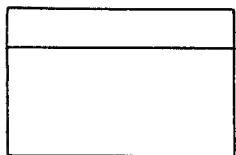
Processor block



Modification block



Decision block



Call to subroutine block



Connector to block within page



Connector to block on another page

Figure 1. Flowchart blocks

## 2.0 GENERAL SYSTEM FLOWCHART NARRATIVES

### A. Regression and Factor Analysis

#### (1) Regression

This program uses three main linkage routines, REGR, COREL, and REGR2. These routines perform functions as follows.

- (a) REGR: This routine reads standard program control cards, and either calls the matrix read subroutine, MXRAD, or reads the data format cards, with FMTRD, and the data cards, with DATRD. In either case, initialization is performed, and if matrix input is called for, and the matrix is a correlation matrix, the link REGR2 is called. If the matrix is the raw sum of squares matrix, the link COREL is called. If card input is the type called for, the raw sum of squares matrix is computed, sequence checks are made on option, the transformation routine TRAN is called on option, sums of observations on variables are computed and the observations as transformed are placed on the disk.

If disk input is desired, the data format cards are not read, sequence check is ignored, but TRAN is called on option. Thus observations previously transformed will be again transformed if the option is taken.

Finally, for card or disk input, the link COREL is called.

- (b) COREL: This routine computes the residual cross products matrix, mean, standard deviations, and correlation matrices. Each of the matrices is printed and/or punched on option by calling the subroutine PRNT. If the raw sums of squares matrix or correlation matrix has been punched, COREL then punches raw sums and sums of squares, or means and standard deviations.

Finally, depending on a switch set in REGR or FCTR, COREL calls the link REGR2 or FCTR1.

- (c) REGR2: REGR2 exits to the monitor if no regression is called for. Otherwise the subroutine REGRE is called, after which an exit is made.

#### (2) Factor Analysis:

This program uses five main linkage routines, FCTR, COREL, FCTR1, FCTR2, and FCTR3, as follows.

- (a) FCTR: Input logic in FCTR is identical with that described under REGR above. On exit, FCTR calls the links COREL or FCTR1, depending on the need for a correlation matrix.
- (b) COREL: Described above under Regression.
- (c) FCTR1: This routine chooses and calculates the communalities, if necessary. Then the subroutines TRIDI and QR are called for eigenvalue computation, after which the link FCTR2 is called.
- (d) FCTR2: After determining the number of factors to be rotated, FCTR2 calls VECTR to compute the eigenvectors. It is at this point that, for the minimization of the number of links required, certain arrays are given a maximum length of ten. The number of eigenvectors computed and the number of factors rotated could exceed ten but for this limitation, which could be eliminated. Eigenvectors are then printed on option, then standardized so that the unrotated factor loadings can be printed, and then communalities are computed and printed.

Finally, if a rotation is called for, link FCTR3 is called; if not, FCTR2 exits to the monitor.

- (e) FCTR3: FCTR3 either calls VARMAX for a varimax rotation or exits to the monitor. If VARMAX is called, then an oblique rotation is performed by calling PROMX or an exit is performed. VARMAX and PROMX use RFOUT for output. Factor scores and regression coefficients are computed on option by calling SCORE. Finally, FCTR3 exits to the monitor.

## B. Analysis of Variance:

This program uses two main linkage routines, ANOVA and ANOV2, which function as follows.

- (1) ANOVA: Standard program control cards are read, followed by the option card. After initialization is performed, card or disk input is chosen. If the data is on cards, FMTRD is used to specify data format, and DATRD to read according to that format. After each card, the data is written on the disk. After transforming (TRAN) on option, the program uses STORE if disk storage is required for the design being analyzed. Finally, link ANOV2 is called.

If the data is to be read from disk, format read is ignored, and the program reads from disk, and calls TRAN and STORE if necessary, before calling link ANOV2.

- (2) ANOV2: This routine calls SDOP, which generates sums and deviates for each factor, MNSQ, which computes component and interaction sums of squares, and REPR, which arranges the analysis of variance table according to the user specified table generation cards. Then the program exits to the monitor.

C. Orthogonal Polynomials: The main linkages for this program are POLY and POL2.

- (1) POLY: After reading all program control cards, POLY chooses disk, cards, or solution vector input. In the case of cards or solution vectors, format read (FMTRD) is called to set up data card format. If input is from cards, data is read by DATRD, and written on the disk. Disk input or card input then is transformed on option, initialization is performed, and if scaling is to be performed, the scaling equation is calculated. Then link POL2 is called.

If solution vectors are to be read, scaling constants and solution vectors are accepted, secondary input (points for polynomial evaluation) is read with DATRD, and link APOL2 is called.

- (2) POL2: This link calls POLSQ unless solution vectors were the input data. POLSQ calculates the orthogonal polynomials and prints them, as well as the solution vectors. If solution vector output is called for, those and the scaling constants are punched by POLSQ.

If the polynomial coefficients are requested, PCOEF is called. If derivatives are required, PDER is called. Finally, PFIT is called if predicted values are desired, after which APOL2 exits to the monitor.

### 3.0 DETAILED FLOWCHART NARRATIVES S

A. Regression Analysis: This program contains three links.

<u>LINK</u>	<u>SUBROUTINES</u>	<u>USE</u>
REGR	Main Program	Inputs parameter cards and source data
COREL	Main Program	Computes correlation matrix
REGR2	REGRE	Computes regression equations

The links communicate with their successors by storing results in common storage.

#### COMMON DATA STORAGE MAP - Regression Analysis

<u>Name</u>	<u>Common Dimension*</u>	<u>Type</u>	<u>Meaning</u>
ICR	1	I	Card read symbolic unit
ICP	1	I	Card punch symbolic unit
IPR	1	I	PRINT-TYPE Switch
ITW	1	I	Output unit numbers
IT1	1	I	Not used
IT2	1	I	Not used
IPROB	1	I	Problem number
N	1	I	Number of variables
NF	1	I	Not used
CASES	1	F	Sum of weights
NPAGE	1	I	Page number
INMD	1	I	Input mode switch
IPRED	1	I	Predicted score switch
ISTEP	1	I	Print steps switch
ICNST	1	I	Pooling switch
IREAR	1	I	Dependent variable
KX	1	I	Not used
MX	20	I	Matrix output options
NCD	3	I	Number of variables on Cards 1, 2, and 3
ISEQ	1	I	Sequence check switch
NCASE	1	I	Number of data cases

NX	10	I	Not used
EFOUT	1	F	Criterion for removing variables in REGRE
EFIN	1	F	Criterion for entering variables in REGRE
TOL	1	F	Tolerance for inverse
FLVB	2	F	Not used
KNN	1	I	REGR or FCTR switch
TITLE	18	F	Page title
VNAME	30	F	Variable names
SUMY	30	F	Summary vector - (Means)
SD	30	F	Summary vector - (standard deviations)
X	30	F	Temporary data vector storage
R	(30, 30)	F	Storage matrix (Correlation)
HIGH	30	F	High value of each variable
HLOW	30	F	Low value of each variable
MF	(50, 3)	I	Variable format storage

- \* The actual number of storage locations occupied by the common variables depends on the variable type. An I, or integer variable, occupies 1 location for each dimension, whereas an F, or Floating Point variable, occupies 2 storage locations.

LINK NAME: REGR  
CALLED BY: // XEQ

This link is used to set common storage with all necessary parameters and data for a multiple regression (REGR). The program begins by reading an input/output units designation card from the card reader. This will store the symbolic units ICR, ICP, IPR, ITW, IT1, IT2. The job-title card, regression card and variable names cards are then read from the symbolic unit ICR and job-title and option cards are printed with verbal designation of their meaning on symbolic unit ITW. If INMD = 1 a variable format card will be read and printed. If NCD2  $\neq$  0 a second variable format card will be read and printed and if NCD3  $\neq$  0 a third format card will be read and printed. Storage and accumulation arrays are initialized and a branch is taken to the appropriate input section determined by the parameter INMD.

If INMD = 1 a data vector containing case identification; card number, weight, field and data elements X (I), I = 1, N where N is the number of variables set by the user, will be read from the card reader and

stored on the disk. If the parameter  $ISEQ \neq 0$  and the  $NCD(I)$  are set to the proper values the input cards will sequence check within case before the elements  $X(I)$  from the card are stored. If  $INMD=2$  the data vector will be transferred from the disk to the core vector  $X$ . When  $INMD=3$  the source data is a matrix and will be read from the card reader, by the subroutine `MXRAD`.

Once the data vector has been transferred from the input device to the core vector  $X$  a test is made to see if the case identification field  $ID1$  is negative or zero. If it is non-positive, the next link (`REGR2` or `COREL`) is read into core storage and executed.

If  $INMD \neq 3$ , the program accumulates the sums vector and sums of squares and cross products matrix from the data vector  $X$ . In addition, the high and low value of each element in  $X$  is also determined. When this information is completed the program branches back to read another data vector.

On exit, all options, heading information, and I/O unit designations are stored in common, along with the summary statistics and cross-product matrix of the input matrix (if the input matrix was a data matrix) or the input matrix itself (if it was a correlation or cross-product matrix). The common variable  $NCASE$  indicates which type of input was accepted.

LINK NAME:    `COREL`

CALLED BY:    `REGR` or `FCTR`

For a description of `COREL`, see Section 3C.

LINK NAME:    `REGR2`

CALLED BY:    `COREL` or `REGR`

The first thing that the link `REGR2` does is test the parameter `IREAR`, which normally contains the column number of the dependent variable,

to determine whether a regression analysis should take place. If  $IREAR = 0$ , subroutine REGRE will not be called and the program will finish with a call exit statement. If  $IREAR$  is greater than zero the subroutine REGRE will be called and the regression equations computed from the correlation matrix, means and standard deviations located in common storage.

### SUBROUTINE REGRE

CALLED BY: REGR2

REGRE performs the following functions:

1. The dependent variable is placed in the last row and column of the correlation matrix  $R$ . That is,  $r_{ij}$  is moved to the last row and column of  $R$ .

Other pertinent vectors are similarly changed.

2.  $r_{i,y}^2/r_{i,i}$  is checked to determine entry variables. If none is entered, REGRE returns to REGR2. Otherwise, requested output is prepared and printed.

3. Entry and exit significance levels are checked, variables for entry or exit are chosen, and output is presented until either degrees of freedom are zero, no more variables are to be entered or removed, or the residual mean square is negative.

B. Factor Analysis: This program will perform a complete factor analysis from either the raw data or a pre-computed correlation matrix.

The factor analysis program contains five links:

<u>LINK</u>	<u>SUBROUTINE</u>	<u>USE</u>
FCTR	Main program	Inputs parameter cards and source data
COREL	Main program	Computes correlation matrix
FCTR1	TRIDI	Tridiagonalizes matrix
	QR	Computes eigenvalues
FCTR2	VECTR	Computes eigenvectors
	COVEC	Solves tridiagonal equations
FCTR3	VARMX	Orthogonal factor rotation
	PROMX	Oblique factor rotation
	SCORE	Computes and outputs factor scores

Each of these links communicates with its successor by storing its results in common storage.

#### COMMON DATA STORAGE MAP - Factor Analysis

<u>Name</u>	<u>Common Dimension*</u>	<u>Type</u>	<u>Meaning</u>
ICR	1	I	Card reader symbolic unit
ICP	1	I	Card punch symbolic unit
IPR	1	I	Print-type switch
ITW	1	I	Printer-typewriter unit
IT1	1	I	Not used
IT2	1	I	Not used
IPROB	1	I	Problem number
N	1	I	Number of variables
NF	1	I	Number of factors
CASES	1	F	Sum of weights
NPAGE	1	I	Page number
INMD	1	I	Input mode switch
IPRED	1	I	Factor score switch

\* The actual number of storage locations occupied by the common variables depends on the variable type. An I, or integer variable, occupies 1 location for each dimension, whereas an F, or floating point variable, occupies 2 storage locations.

ICOM	1	I	Communality option
IROT	1	I	Rotation switch
NFRT	1	I	Number of factors to rotate
KX	1	I	VARMX/PROMX switch
MX	20	I	Matrix output options
NCD	3	I	Number of variables on Cards 1, 2, and 3
ISEQ	1	I	Sequence check switch
NCASE	1	I	Number of data cases
KCNT	1	I	Parameter for factor count
KNN	1	I	REGR or FCTR Switch
NX	9	I	NX(1) is a pooling switch
TRC	1	F	Trace
FLVB	4	F	Not used
TITLE	18	F	Page title
VNAME	30	F	Variable names
SUMY	30	F	Summary vector (Means)
SD	30	F	Summary vector (Standard deviations)
X	30	F	Temporary data vector storage
R	(30,30)	F	Storage matrix(Correlation)
HIGH	30	F	High value of each variable
HLOW	30	F	Low value of each variable
MF	(50, 3)	I	Format storage
ALPHA	30	F	{ Contain elements of tridiagonal matrix
BETA	30	F	

LINK NAME : FCTR

CALLED BY: //XEQ

The first link loaded is FCTR which reads all parameter cards and stores the analysis options and parameters in common storage. Then either a pre-computed matrix is read or a raw cross product matrix is formed from the raw data matrix. The common variable NCASE is set to either a negative or positive value depending on whether a correlation matrix or input data was read. Link FCTR1 is then loaded into core if NCASE is less than zero; otherwise link COREL is loaded.

LINK NAME: FCTR1

CALLED BY: LINK FCTR

This link is used as a factor analysis setup program. From the parameter ICOM, which has been determined by the user, the diagonal elements of the correlation matrix are replaced by estimates of the communalities. There are three possible values of ICOM and these correspond to the three primary branches in the program.

If ICOM = 0, the diagonal elements of the correlation matrix are unchanged. In effect, this corresponds to a principal components analysis where the communality estimate is equal to 1.

If ICOM = 1, each element on the diagonal will be replaced by the absolute value of the largest off-diagonal element in a row.

If ICOM = 2, each diagonal element will be replaced by the square of the multiple correlation coefficient (i.e., if  $i$  represents the  $i$ th diagonal element, then  $R_{i\cdot}$  will be the multiple correlation between the  $i$ th variable and all other variables in the matrix).

After the communality estimates have been determined, the program computes the trace of the matrix by summing the diagonal elements and storing the result at the symbolic location TRC. The subroutines TRIDI and QR are then called to compute the eigenvalues of the new matrix.

Upon entry to the program, the correlation matrix, or matrix to be factored, is assumed to be located in the matrix R. The parameters N and ICOM have been read into common storage by link FCTR.

When link FCTR2 is called the trace of the correlation matrix is in location TRC, the diagonal elements of the tridiagonalized correlation matrix are in array ALPHA, the off-diagonal elements are in array BETA and the eigenvalues are in array X.

SUBROUTINE NAME: INVRS

CALLED BY: FCTR1

Description: INVRS inverts a symmetric matrix with unit diagonal

elements. On entry, the matrix is in array R. The upper triangular part of the matrix is replaced by the elements of the inverse. The part below the diagonal is not modified.

SUBROUTINE: TRIDI

CALLED BY: FCTR1

This subroutine converts a symmetric matrix to tridiagonal form.

The method employed is Householder's method. In this method, N-2 elementary orthogonal transformations are chosen in such a way that the transformation will leave only the first subdiagonal element in the rth column nonzero. The final matrix

$$A' = P_{n-2} P_{n-3} \cdots P_2 P_1 A P_1 P_2 \cdots P_{n-3} P_{n-2}$$

can be stored in two arrays, the first (ALPHA) containing the elements of the main diagonal and the second (BETA) containing the first sub-diagonal element in each column (except the last).

Along with the transformed matrix a transformation matrix is formed and stored over the original matrix.. This matrix is computed as

$$T = P_1 P_2 P_3 \cdots P_{n-2}$$

and has the property that an eigenvector of the tridiagonal matrix, when premultiplied by T, becomes an eigenvector of the original input matrix.

On entry, the correlation matrix is in array R.

On exit, the transformed matrix is in common arrays ALPHA and BETA. The transformation matrix is in R. The infinity norm of the transformed matrix is in the common cell ANORM.

SUBROUTINE    QR

CALLED BY:   FCTR1

QR finds up to thirty eigenvalues of the tridiagonal matrix previously prepared by the routine TRIDI. The QR method is used.

On exit, the eigenvalues are in descending order in the vector X.

LINK NAME:   FCTR2

CALLED BY:   LINK FCTR1

This subroutine is used to compute and output the factor matrix. First it determines the number of factors to compute from the parameters NF and KCNT. If NF = 0 then the factors computed will be those which have characteristic values greater than or equal to 1. If NF = 2 then the number of factors computed will be the number which is in KCNT. If NF = 3 the number of factors computed will be only those factors which account for KCNT percentage of the trace.

As each characteristic value is examined a cumulative sum is computed and the cumulative percentage of trace is computed.

The routine VECTR is called to compute the required number of eigenvectors, and the factor matrix is computed by the expressions:

$$R(I, J) = R(I, J) * \text{SQRT}(X[J])$$

where R(I, J) on the right side of the equal sign contains the characteristic vector and X(J) is the Jth characteristic value.

Once the factor computation has been completed the characteristic values and cumulative percent of trace are printed along with the title, page number, trace, sum of the characteristic values, and the difference between the trace and sum. Subroutine PRNT is then called to output the factor matrix. Communalities are calculated from the sums of squares of the row elements of the factor matrix, and printed.

When the link is called the elements of the tridiagonal matrix are in arrays ALPHA and BETA, and the characteristic values are in array X. The parameters NF and KCNT are at values assigned by the user.

Upon exit from the link R contains the principal axis factor matrix.  
The parameters NF and KCNT have been changed to the following values:

<u>Entry</u>	<u>Exit</u>	
NF	<u>NF</u>	<u>KCNT</u>
0	0	0
1	0	0
2	KCNT	0
3	0	KCNT

SUBROUTINE NAME: VECTR

CALLED BY: FCTR2

VECTR is a subroutine that computes NF eigenvalues of an N x N matrix by computing the eigenvalues of the tridiagonalized matrix obtained by subroutine TRIDI and transforming them to the characteristic vectors of the original matrix via a transformation matrix. (See TRIDI narrative.)

The method by which the K<sup>th</sup> eigenvector is found is as follows:

The eigenvector array (V) is initialized to ones as a first approximation. Subroutine COVEC is called to compute

$$Q = (A - X_k I)^{-1} V$$

where A is the tridiagonalized matrix and X<sub>k</sub> is the k<sup>th</sup> eigenvalue. If V and Q, when normalized, do not agree (element for element) within .05, V is set equal to Q and the routine reiterates. If V and Q agree, the vector

$$R = V - (A - X_k I) Q$$

is formed and COVEC is called to compute

$$Y = (A - X_k I)^{-1} R.$$

Z = V + Y is then the eigenvector of the tridiagonal matrix. Z is then normalized and premultiplied by the transformation matrix and written onto the disk.

When all NF eigenvectors have been found and written on the disk, they are read back in over the transformation matrix.

The tridiagonalized matrix must be in arrays ALPHA and BETA, its eigenvalues must be in array X, and the transformation matrix must be in the Matrix R, on entry.

On exit, the eigenvectors are in R and the transformation matrix is destroyed. If a rotation is required, FCTR3 is called.

SUBROUTINE NAME: COVEC

CALLED BY: VECTR

COVEC solves the system of tridiagonal equations

$$(A - XI) \cdot V = C$$

for V, where A is a tridiagonal matrix of the form

$$\begin{bmatrix} a_1 & b_2 & 0 & 0 & 0 & . & . & 0 \\ b_2 & a_2 & b_3 & 0 & 0 & . & . & 0 \\ 0 & b_3 & a_3 & b_4 & & & & \\ 0 & 0 & b_4 & a_4 & & & & \\ 0 & 0 & 0 & . & & & & \\ . & . & . & & . & 0 & & \\ . & . & . & & & . & b_n & \\ 0 & 0 & . & . & 0 & b_n & a_n & \end{bmatrix}$$

which was stored in the arrays ALPHA and BETA. X is an approximate eigenvalue of A, and C is a column vector. An eliminative scheme is used which uses the largest element in each column as its pivot element.

The arguments used when calling this routine have the following meaning:

CONS	= Vector or right side of equation
VECT	= Vector to be solved for

In addition, the program requires the arrays ALPHA and BETA.

The solution vector is in the argument VECT, on exit.

SUBROUTINE NAME: PRNT

CALLED BY: FCTR2

This subroutine is described in Section 3C.

LINK NAME: FCTR3

CALLED BY: FCTR2

This link calls VARMX if an orthogonal rotation is required, PROMX if an oblique rotation is required, and SCORE for factor score calculations. RFOUT is used for output, and MATIN for matrix inversion. FCTR3 then exits to the monitor.

SUBROUTINE NAME: VARMX

CALLING PROGRAM : FCTR3

After initializing NFRT, the number of factors to be rotated, and setting the tolerance EPS, the program sets the B matrix to an identity matrix. The A matrix, which contains the factors to be rotated, is then row normalized by dividing each row element by the communality for that row.

The main iteration cycle is then initiated by computing a convergence criterion and comparing it to the criterion on the previous cycle. If it is approximately zero, control will be returned to calling program. If greater than zero it initiates a new cycle. A cycle consists of a pairwise

rotation of the factor matrix. The program determines the sine and cosine of the angle of rotation and proceeds to apply this angle to the matrix. However, if this angle is less than 1 minute of 1 degree (EPS) then a rotation will not be effected.

After the factor matrix has been rotated by the sine and cosine of the rotating angle, the B matrix, which initially contains the identity matrix, is also rotated by this angle. The program then begins another iteration cycle. At the beginning of each iteration the cycle count and convergence criterion are printed. A test is made to determine if more than 50 cycles have taken place. If so the program will terminate.

Upon entry to the program the factor matrix is located at A, the number of factors to be rotated is contained in NFRT. If this field is zero, the program will set it equal to the number of factors as determined by the program FCTR2.

Upon exit from the program the array A contains the orthogonally rotated matrix and B contains the transformation matrix.

SUBROUTINE NAME: PROMX

CALLING PROGRAM: FCTR3

This subroutine, in conjunction with RFOUT, is used to perform an oblique rotation of a factor matrix.

After setting IAL to four, the program computes the inverse of  $A^T$ .  $A = B$  where A is the orthogonal factor matrix and  $A^T$  its transpose.

Row- and column-normalizing vectors H and G are then computed for use in the computation of the E matrix. The expression for this is:

$$E = \dot{A}^T * P$$

where P = row normalized A matrix with each element raised to the IALth power. The sign of each element remains the same as the unpowered element.

Following this the transformation matrix to the oblique reference

vector structure matrix is computed by:

$$B = B * E$$

where B on the right contains  $A^T \cdot A$ .

The transformation matrix is then formed by normalizing the columns of B.

Once this transformation matrix is complete, it is applied to the A matrix to form the reference vector structure matrix. Also, multiplying it by its transpose produces the correlations among reference vectors.

Upon entry to the program, array A contains the orthogonally rotated factor matrix from VARMX.

Upon exit, A contains the oblique reference vector structure matrix, B contains the transformation matrix and E contains the correlations among reference vectors. The common variable KX(1) has been set equal to 1 for program RFOUT.

SUBROUTINE NAME: RFOUT

CALLING PROGRAM: FCTR3

This subroutine is used to output the results of an orthogonal rotation and/or compute and output the remainder of the matrices associated with an oblique rotation. The program determines whether the program preceding it was VARMX or PROMX by the common variable KX(I). If KX(I) = 0 then VARMX preceded and RPRNT is called to output the transformation matrix and the orthogonal factor matrix. Before returning to the calling program, B is set to an identity matrix for possible factor score computation.

If KX(I) = 1 then the preceding program was PROMX and different output and computational functions are performed by RFOUT. RPRNT is called to output the correlations among oblique reference vectors, (E), oblique reference vector structure matrix, (A), and the correlations among oblique reference vectors, (B). Matrix E is then inverted by MATIN and the reference vector pattern matrix computed and RPRNT

called to output this matrix. The correlations among reference vectors and primary factors are then computed and printed by RPRNT. Using this result the correlations among primary factors are computed and presented. Finally, the primary factor structure matrix and primary factor pattern matrix are computed and presented.

Upon entry  $KX(I) = 0$  if entry was from VARMX and A contains the orthogonal factor matrix, B the transformation matrix.

When  $KX(I) = 1$ , A contains the oblique reference vector pattern matrix, B contains the transformation matrix and E contains the correlations among reference vectors.

Upon exit from the program A will contain the primary factor pattern matrix and B, if from VARMX, will contain an identity matrix, or, if from PROMX, will contain the correlations among primary factors.

SUBROUTINE NAME: RPRNT

CALLING PROGRAM: RFOUT, SCORE

This subroutine is used to print the following matrices:

1. Orthogonal transformation matrix
2. Orthogonal factor matrix
3. Transformation to oblique reference vector structure
4. Oblique reference vector structure
5. Correlations among oblique reference vectors
6. Oblique references vector patterns
7. Correlations between reference vectors and primary factors
8. Oblique primary factor structure
9. Correlations among oblique primary factors
10. Oblique primary factor loadings
11. Factor score regression coefficients

This program has the same logic and structure as subroutine PRNT except for two minor differences. Column headings on printout are numerical sequence values and are not taken from the array VNAME. The second major difference is the meaning of the argument CODE. RPRNT will output either the common array A or B; if  $CODE = 0$ , then A will be printed with row headings taken from VNAME and columns in

generated numerical sequence. If  $KODE = 1$  then the array B will be printed with generated numerical sequence for column and row headings.

Matrix B or E contains the output matrix if  $KODE = 1$  or the A array contains the output matrix if  $KODE = 0$ . MID contains the matrix identification number,  $KODE = 1$  or 0. NR is the number of columns of the output matrix. These are the exit conditions.

SUBROUTINE NAME: MATIN

CALLING PROGRAM: RFOUT, SCORE

MATIN inverts a symmetric matrix.

SUBROUTINE NAME: SCORE

CALLING PROGRAM: FCTR3

SCORE is used to compute the factor score regression coefficients and factor scores from either an oblique or orthogonal factor matrix. The program divides each element in the factor matrix A by  $1 - X(I)$  where  $X(I)$  are the communalities for each row. The resultant matrix is stored in the last ten columns of the array A.

The transpose of the original matrix multiplied by this matrix is then added to the B matrix. The B matrix contains either an identity matrix if the factors are orthogonal or the inverse of the intercorrelations of the primary factors and oblique factors. The resultant matrix B is then inverted by subroutine MATIN and the inverse multiplied by the modified A matrix to form the factor score regression coefficients. Subroutine RPRNT is then called to output this matrix.

Factor scores are then computed from the regression coefficients by reading a data factor from the disk. If the problem number ID is positive, each variable  $X(I)$  in the data vector is standardized by :

$$Z(I) = (X[I] - \text{SUMY}[I]) / \text{SD}(I)$$

where SUMY(I) contains the mean of the ith variable and SD(I) contains the standard deviation. The N elements of the standardized data vector Z are then multiplied by the N elements in each of the NFRT regression coefficients in A to form the factor scores for this data vector. The vector is then printed out with a sequence number and case identification ID. The title, job number and column headings are also printed on each page.

If ID is negative the program will terminate and return to the main calling program.

Upon entry to the subroutine matrix A contains the factor matrix. The raw data, followed by an artificial data vector with a negative identification must be located on the disk. Matrix B will contain an identity matrix if the factors are orthogonal or the primary factor intercorrelations if the factors are oblique. The arrays SUMY and SD contain the means and standard deviations respectively.

Upon exit the array A contains the factor score regression coefficients and the factor scores have been printed and/or punched.

SUBROUTINE NAME: XMAX

XMAX computes the maximum of two arguments.

### C. Routines Used by Regression and Factor Analysis

LINK NAME: COREL

CALLING PROGRAM: REGR and FCTR

After initialization of switches and moving the sums of squares from the diagonal elements of the cross product matrix R to the vector SD for possible punchout, subroutine PRNT is called to examine MX(1) for either printing and/or punching the raw cross products matrix. From the raw cross products matrix, the residual cross products matrix is then computed by:

$$R(I, J) = R(I, J) - \text{SUMY}(I) * \text{SUMY}(J) / \text{CASES}$$

where: R(I, J) on the right hand side of the equal sign contains  
I, Jth raw cross products

SUMY(I) contains the raw sum

CASES contains the number of observations

After the computation is completed, subroutine PRNT is then called for printing and/or punching.

From the residual cross product matrix, the variance-covariance matrix is computed by

$$R(I, J) = R(I, J) / (\text{CASES} - 1)$$

where: R(I, J) on the right contains the residual cross products and  
and CASES contains the number of observations.

After the computation is completed, subroutine PRNT is again called for possible output.

Once the variance-covariance matrix is computed the means and standard deviations are computed by:

$$\text{SUMY}(I) = \text{SUMY}(I) / \text{CASES}$$

$$\text{SD}(I) = \text{SQRT}(R[I, I])$$

A summary statistics table is then printed which contains the number of cases, variable names, high and low value of each variable, and

means and standard deviations.

Once this printout is completed, the correlation matrix is computed by:

$$R(I, J) = R(I, J) / (SD[I]*SD[J] )$$

In the computation a test is made to determine if either SD(I) or SD(J), the standard deviation of the Ith and Jth variable respectively, is zero. If either one is zero, the correlation coefficient R(I, J) is set to zero and a message indicating which variable has the zero variance is printed.

After the computation is completed, subroutine PRNT is again called to output the matrix.

Upon entry to the program, CASE (the number of observations), SUMY (the cumulative raw sums of each variable), and R(I, J) (the cumulative raw cross product matrix) have been either read in as matrices or accumulated previously. The high and low values of each variable are also present in the vectors HIGH and HLOW.

Upon exit from the program the means, standard deviations, correlation matrix and sum of weights are in common storage at locations represented by SUMY, SD, R, and CASES, respectively.

SUBROUTINE NAME: PRNT

CALLING PROGRAM: COREL, FCTR2

Subroutine PRNT is used to print and/or punch the following six matrices:

1. Matrix of raw cross products
2. Matrix of residual cross products
3. Variance-covariance matrix
4. Matrix of correlation coefficients
5. Matrix of characteristic vectors
6. Principal axis factor matrix

The program examines the output option array MX subscripted by the argument MID. If MX(MID) = 0, control will be returned to the calling program and no output will occur. If MX(MID) = 2 or 3, control will

be transferred to the punch routine and the matrix will be punched, 5 elements to a card with identification indicating the problem number (IPROB), the matrix identification number (MID), the row of the matrix in which these 5 elements are located (I), and the column of the first element on the card (K).

After the matrix has been punched MX(MID) is again examined to determine if it contains a value of 1 or 2. If it does not, the program will return to the calling program. However, if it does contain 1 or 2, the program will branch to the print routine. The print routine will print the title, page number, and job number followed by the name of the matrix as identified by MID. The matrix is printed, 8 elements to a line, with each column and row identified by a variable name as contained in VNAME. After the entire matrix is printed, control is returned to the calling program.

There are four arguments used in the calling sequence to the subroutine. These have the following meaning:

MID	Matrix identification number
KODE	KODE is unused, but could be used for a switch allowing different formats.
NR	contains the number of rows in the matrix
NC	contains the number of columns in the matrix

The matrix to be printed or punched is located in the common array R.

On exit, the common variable NPAGE has been incremented by the number of pages required to print the entire matrix. There are no other changes to any other common locations.

SUBROUTINE NAME: MXRAD

CALLING PROGRAM: FCTR, REGR

This subroutine is used by link FCTR and link REGR to read and/or add matrices. The program starts by reading a card containing the problem number(IP), matrix identification number (MID), the column number of the first data element on the card (IC), the row number (IR), and 5 data elements X(I), I = 1, 5. If IP is negative, the program branches to a termination routine which will set the common variable

NCASE to either a positive or negative value depending on whether the correlation matrix was processed. Control is then transferred to link FCTR or link REGR.

If IP is positive and the row card is from a matrix other than 21 or 22, the 5 elements X(I) are added to contents of the core matrix R, subscripted by IC and IR. If MID = 21, there is only one legitimate element on the row card, and this, added to the common variable CASES, is the number of observations. If MID = 22, there are only two legitimate elements on the row card, and these are added to the common vectors SUM and X.

After the row card has been added to a matrix or vector (R, CASES, SUM or X) another card is read and the same process is initiated. Cards will be read until a negative problem number card comes up and the process is terminated, unless ICNST is non zero. If this is the case, a second matrix is accepted, and subtracted from those previously read. In this case, matrices should be raw cross products matrices.

Upon entry to the program, the common variables R, NCASE, CASES, SUM and X have been set to zero. The variable ICNST is set to allow pooling.

Upon exit, R, CASES, SUM, X have been set with input from the card reader. The variable NCASE has been made either positive or negative to determine logic flow in the main calling program. NCASE positive implies a raw cross products data set has been read and control will be passed to the correlation matrix generation program COREL. If NCASE is negative or zero, control will bypass COREL as a correlation matrix data set has been read.

# D. Analysis of Variance

<u>LINK</u>	<u>SUBROUTINES</u>	<u>USE</u>
ANOVA	Main Program	Inputs parameter cards Inputs source data
ANOV2	SDOP	Forms sums and deviates
	MNSQ	Forms sum of squares
	REPRT	Forms mean square and output table.

## COMMON DATA STORAGE MAP - Analysis of Variance

<u>Name</u>	<u>Common Dimension*</u>	<u>Type</u>	<u>Meaning</u>
ICR	1	I	Card reader symbolic unit
ICP	1	I	Card punch symbolic unit
IPR	1	I	Print-Type switch
IT1	1	I	Not used
IT2	1	I	Not used
IPROB	1	I	Problem number
NPAGE	1	I	Page number
INMD	1	I	Input mode
NF	1	I	Number of factors
ITRN	1	I	Transformation switch
NA	1	I	Number of levels +1, Factor 1
NB	1	I	Number of levels +1, Factor 2
NC	1	I	Number of levels +1, Factor 3
ND	1	I	Number of levels +1, Factor 4
TITLE	18	F	Page title
NX	5	I	Number of levels for each factor
LS	5	I	Temporary constants
IN	4	I	Data input array
NDIV	20	I	Divisors for sum of squares
SMQR	20	I	Summary vector for sum of squares
XDEV	20	I	Storage for deviates
X	1500	F	Data storage array
ITW	1	I	Output unit numbers

\* The actual number of storage locations occupied by the common variables depends on the variable type. An I, or integer variable, occupies 1 location for each dimension, whereas an F, or floating point variable, occupies 2 storage locations.

LINK NAME: ANOVA

LOADED BY: // XEQ

This link is used to read parameter cards and source data for analysis of variance. The program first reads an input-output units designation card from the card reader. It then reads a title card and the analysis of variance parameter card. If the parameter INMD = 1 a variable format card is read and printed.

After initializing the storage parameters for the number of factor levels the program reads a data record from either the card reader if INMD = 1, or from the disk if INMD = 2. The data record contains an index array IN and a data item. If the first index array item IN(1) is positive, the program will compute the storage location IS for this data item from the index array IN and the storage parameter LS. If the transformation switch is on, the transformation program will be called. Following this, the STORE program will be called to either store the data item DATA in storage or on disk. Following the return from program STORE, the program will branch back to read another data record.

Upon exit from the program all the parameters are in common and all the required data has been stored either on the disk or in the array X. The condition for storing the data in X is determined by the storage parameter IS. If IS is greater than 1500 the data will be stored on the disk; otherwise, in the array X.

SUBROUTINE NAME: STORE, GET

CALLING PROGRAM: ANOVA, SDOP, MNSQ

These subroutines are used to store or get data either from the array X or disk. The programs test the argument IS; if IS is less than 1500 the data will be stored or retrieved from the core array X. If IS is greater than 1500 the item will be stored on the disk at storage location IS-1500. After the item has been either stored or retrieved, control is returned to the calling program.

On entry, DATA contains the item.  
IS contains the location parameter.

On exit,

STORE - DATA has been stored in X or on the disk.

GET - DATA has been retrieved from X or from the disk.

LINK NAME: ANOV2

CALLED BY: ANOVA

This program calls the remaining analysis of variance programs, SDOP, MNSQ, and REPR, and exits to the monitor.

SUBROUTINE NAME: SDOP

CALLING PROGRAM: ANOV2

This subroutine is used to generate the analysis of variance sums and deviates for each factor. The program computes appropriate storage locations for the data and calls subroutine GET to obtain the data item for the Kth factor from either the array X or from the disk. Each data item is then summed over all levels for this factor and the sum located at SUMX is stored back in the array X at the appropriate location IS.

After the sum is computed for the Kth factor the data array is again used to compute the analysis of variance deviate. Each element used to form SUMX is replaced by

$$DATA = FN * DATA - SUMX$$

where FN is the number of levels in the factor.

After this computation, the storage pointers IS and ISPM are incremented and a test is made to determine the appropriate level. The factor count K is then incremented and computations are performed on the transformed data elements. After passing through the data the program returns to the main calling program

Upon entry to the program all data items have been stored in either

the array X or on the disk. The number of levels for each factor is located in the array NX and the number of factors is located in NF.

Upon exit from the program the data array (either X or disk) contains the sums and analysis of variance deviates.

SUBROUTINE NAME: MNSQ

CALLING PROGRAM: ANOV2

This subroutine is used to compute the component and interaction sums of squares for the final analysis of variance table. After initialization of the cumulation arrays, the program determines which component in the analysis of variance table is to be incremented for the current data item. The analysis of variance table SMQR can contain at most 15 values. These are related to the component and interaction sum of squares as follows:

<u>Index</u>	<u>Component</u>
1	A
2	B
3	C
4	D
5	AB
6	AC
7	AD
8	BC
9	BD
10	CD
11	ABC
12	ABD
13	ACD
14	BCD
15	ABCD

where A, B, C, D are names of the factors. It should be noted that even if a particular job does not involve four factors, the subscript for the particular component is still the same.

By passing through the data array (core and/or disk) in a sequential

manner, the program is able to determine which index value is required for SMQR by testing the individual factor level counts IA, IB, IC, ID and comparing these to the number of levels in each factor, NA, NB, NC, ND. When the proper subscript is determined, K is set equal to this value and the program adds the square of the deviate to the appropriate cell in SMQR. When all deviates have been processed the program returns to the main calling program.

On entry, the analysis of variance deviates are located either in the X array and/or on the disk. The number of levels in each factor are located in NA, NB, NC and ND.

On exit, the component and interaction sums of squares multiplied by the component or interaction are located in SMQR. The deviates of interest are in XDEV and the divisor necessary to obtain the component sum of squares is located in NDIV.

SUBROUTINE NAME:   REPRT

CALLING PROGRAM:   ANOV2

This program is used to output the analysis of variance table. The program begins by setting up a general array for the degrees of freedom. Next, the appropriate divisor and accumulation arrays are initialized, and the total sum of squares is computed. A card, containing a 24-character row heading (HEAD), a control indicator (INDI), and a component summary index array (INX), is then read from the card reader. The index array, INX, is then used to subscript the SMQR array, which contains the component sums of squares. To add the appropriate elements to form the component to SMSQ and degrees of freedom NDF1 after all elements of INX are chosen, the mean square SMSQM is computed by dividing the sums of squares by the degrees of freedom. Once this computation is completed, a line is printed containing the sums of squares, degrees of freedom and mean square. If INDI is greater than zero, a page will be skipped and a title line with column headings will be printed before the component line. If INDI is negative the program will terminate by printing a residual line if necessary and/or total line. The residual is the difference between the total sum of squares computed in the beginning of the program and the sum of squares accumulated after each line has been printed.

On entry, except for the proper divisor, the component sums of squares are located in SMQR, and NDIV contains the divisor to compute the sums of squares.

On exit, SMQR contains the component sums of squares and the requested component lines have been printed.

## E. Orthogonal Polynomials

The program contains the two links:

<u>LINK</u>	<u>SUBROUTINES</u>	<u>USE</u>
POLY	Main Program	Inputs parameter cards and source data
POL2	POLSQ	Determines degrees and computes orthogonal polynomials
	PCOEF	Computes coefficients of fitted polynomial
	PDER	Computes derivatives at a point
	PFIT	Computes predicted Y for a given X.

## COMMON DATA STORAGE MAP - Orthogonal Polynomials

<u>Name</u>	<u>Common Dimensions*</u>	<u>Type</u>	<u>Meaning</u>
ICR	1	I	Card reader symbolic unit
ICP	1	I	Card punch symbolic unit
IPR	1	I	Print-Type switch
ITW	1	I	Output unit numbers
IT1	1	I	Not used
IT2	1	I	Not used
IPROB	1	I	Problem number
N	1	I	Maximum degree of polynomial
NF	1	I	Actual degree of polynomial
CASES	1	F	Not used
NPAGE	1	I	Page number
INMD	1	I	Primary input mode
ISCR	1	I	Predicted values switch
NCASE	1	I	Number of data cases
ICOF	1	I	Coefficient computation switch
IDER	1	I	Derivative computation switch
NDER	1	I	Order of derivatives
IALP	1	I	Polynomial solution vector output switch
INMD2	1	I	Secondary input mode
KX	3	I	Not used
EPS	1	F	Tolerance criterion

FLVB	4	F	Not used
XB	1	F	Scaling Constant
X14	1	F	Scaling Constant
TITLE	18	F	Page title
ID	150	F	Identification codes
X	150	F	Values of X
Y	150	F	Values of Y
C	51	F	Polynomial solution vector
ALPHA	51	F	Polynomial solution vector
BETA	51	F	Polynomial solution vector
MF1	50	I	Format for input data

\* The actual number of storage locations occupied by the common variables depends on the variable type. An I, or integer variable, occupies 1 location for each dimension, whereas an F, or floating point variable, occupies 2 storage locations.

LINK NAME: POLY

LOADED BY: // XEQ

This program is used to read parameter cards and source data for orthogonal polynomials. The program first reads an input-output units designation card from the card reader, followed by a title card and the orthogonal polynomials parameter card. If the parameter INMD = 1 or 3, a variable format card is read and printed. The program then branches to a special input section for each value (1, 2, or 3) of the parameter INMD.

If INMD = 1 the program will read the source data from the card reader. Each input record contains an identification field (ID[I]), a derivative computation indicator (IDR), an X value X(I), and a Y value Y(I). If ID(I) is positive, the program will test IDR for non-zero. If zero, the program will read another card record; if non-zero, the identification for this record ID(I), will be made negative for examination by the program PDER. If ID(I) is negative, link POL2 is loaded and executed.

If INMD = 2, the input data will be read from the disk instead of the card reader. It was placed there by the previous use of INMD = 1.

If INMD = 3, the polynomial solution vectors will be read into arrays

ALPHA, BETA, and C respectively, along with any necessary scaling constants. A branch is then made to the section corresponding to INMD = 1 in order to read the data points.

Upon exit, the analysis parameters and data points are in common. In addition, if the parameter INMD2  $\neq$  0 the polynomial solution vectors are in COMMON. If scaling was requested, scaling constants are also in COMMON.

LINK: POL2

CALLED BY: POLY

POL2 calls the remaining programs in this analysis type if they are required, i.e., POLSQ for polynomials, PCOEF for coefficients, PDEF for derivatives, and PFIT for evaluation and prediction.

SUBROUTINE NAME: POLSQ

CALLING PROGRAM: POL2

After initializing the computational parameters and accumulation arrays the program begins the main iteration loop by computing the first polynomial solution vector C by:

$$C(II) = S/RO$$

where II is the current degree of the computed orthogonal polynomial,  
S is the inner product of Y and IIth degree orthogonal polynomial,  
RO is the inner product of the polynomial with itself.

Once S is computed the cumulative predicted values for 1, 2..IIth degree polynomials are computed and stored in array YA. The variance criterion for the cycle is then computed and compared to its value on the previous cycle. If the difference is approximately equal (within the tolerance EPS) it will transfer to the output routine and return to the main calling program.

If the variance criteria are not equal the next order polynomial will be computed utilizing the next order solution vectors ALPHA and BETA. After each order polynomial has been computed it is stored in the array POL. A test is made to determine if four polynomials have been stored. If so, the array POL is printed along with the input values contained in ID, X and Y. Also printed for each X(I), Y(I) are the cumulative predicted values from YA(I) and their difference.

The title information, job number, page number, and column headings are printed at the top of each output page. The current solution vectors are also printed at the bottom of each page.

At the conclusion of the output stage, the current variance criterion is stored in the previous criterion location and the polynomial order II is incremented. The program then branches back to initiate another cycle.

After either the variance criterion has been satisfied or the maximum degree of the polynomial (as determined by the user) has been reached, the program tests the input parameter IALP to determine if the final solution vectors are to be punched. If punching has been requested, the vectors are punched, with a matrix identification number, row and column number in the standard matrix punch output format. Also, necessary scaling constants are punched.

Upon entry to the program the data is stored in array X and Y. The number of data cases are in location NCASE and all necessary common parameters are located in COMMON storage.

On exit from the program, the solution vectors ALPHA, BETA and C are located in COMMON storage and the degree is that of the resultant polynomial which either satisfied the variance criterion or is the input parameter N which is located at NF. Arrays X, Y, ID and location NCASE have not changed.

SUBROUTINE NAME: PCOEF

CALLING PROGRAM: POL2

This subroutine is used to compute the coefficients of the fitted polynomial from the polynomial solution vectors.

After initialization, the program computes orthogonal polynomials using the solution vectors ALPHA and BETA. From the orthogonal polynomial the coefficients of the fitted polynomial are computed by multiplying the solution vector C by this polynomial. This process continues until the degree NF+1 is reached.

After the computation is completed, the coefficients are printed with title and heading information.

Upon entry to the program the solution vectors are contained in arrays ALPHA, BETA and C. The degree of the polynomial is located at NF.

Upon exit from the program, the polynomial solution vectors ALPHA, BETA and C are in common storage and the degree of the polynomial is at location NF.

SUBROUTINE NAME: PDER

CALLING PROGRAM: POL2

This subroutine is used to compute the derivative of the fitted polynomial at a given point. The program begins by examining the identification vector ID for a negative value. If ID(I) is positive, I is incremented and another identification value is examined. This will continue until I is equal to NCASE in which case control will be returned to the main calling program.

If, for a given I, ID(I) is negative, the value of X for this I will be stored in XB and other derivative computations for this point started by initializing the computational arrays and parameters. The parameter NN represents the order of the derivative to be computed and is initially set equal to 1.

The program then computes the NNth order derivative by utilizing the polynomial solution vectors ALPHA, BETA and C to compute the orthogonal polynomial DOPOL.

As each order polynomial is computed a recurrence solution is utilized to build up the value of the derivative and the next order. When NN is equal to NDD1, the order of the requested derivative, the program will print line or lines containing the identification ID(I), the value of

X(I), the value of Y(I), the order of the derivative and its value. Each page will also contain title and column headings.

After the derivative for a point has been printed the program will transfer back to examine another ID(I) for a negative value.

Upon entry to the program, the array ID contains identification values, X, Y contain data and ALPHA, BETA and C contain the polynomial solution vectors. NF contains the degree of the polynomial, NCASE the number of data points and NDER the order of the derivatives to be computed.

Upon exit from the program the derivatives for all points indicated in the ID vector have been printed, and the polynomial solution vectors ALPHA, BETA and C are in their respective arrays.

SUBROUTINE NAME: PFIT

CALLING PROGRAM: POL2

This subroutine is used primarily to compute predicted values from a set of data values X(I) that are different than those used to compute the initial polynomial. After initialization the program uses the solution vectors ALPHA, BETA and C to compute orthogonal polynomials.

As each order orthogonal polynomial is generated the cumulative predicted value is computed from X(I) and stored in the array YA. After NCASE values of YA are computed the program will print the predicted values, with identification, the actual value of Y, and the difference. Title and column headings will also appear on each page.

Upon entry to the program the solution vectors are in ALPHA, BETA and C. The data points are located in X and Y and the degree of polynomial is in NF. The number of data points is located at NCASE.

Upon exit from the program the predicted values for all data points have been printed.

## F. Routines Used by All System Programs

The following five routines in assembly language allow user-specified format statements at object program time. Of the routines called by these, CARDZ, PRNTZ, NORM, IFIX, TYPE Z, and FSTOX are utility routines available to the assembler.

### SUBROUTINE FMTRD

FMTRD reads one card containing a format and stores it in a form suitable for the subroutine DATRD.

Calling sequence:

CALL FMTRD (FORMAT, ERROR)

FORMAT must be an integer vector fifty (50) words long. ERROR is an integer word.

Upon return, FORMAT contains the translated format and ERROR will be zero. If the translation was completed, ERROR will be the next column to be processed if an error was detected. When an error is detected no attempt is made to complete the translation and format may have to be changed.

Format codes: The following specifications are acceptable:

wX      nIw      nFw.d      nEw.d

n may be omitted if it is one. One level of parentheses is allowed for group repetition. In addition, parentheses are required around the entire format. Every specification, including wX and parenthesized groups, must be followed by either a comma or a right parenthesis. Multiple record formats (/), scaling (P) and alphabetic conversion (A, H and I) are not available. In addition, the format must be completed on one card.

Length: 225

Subroutines required: CARD Z

### SUBROUTINE PRNTB

PRNTB prints the I/O buffer after a previous read statement.

Calling sequence:

CALL PRNTB

Function and use: When called, PRNTB prints the first eighty positions of the I/O buffer on the printer with a double space. It may be used after a call to FMTRD or DATRD, whether or not an error occurred, to print the card just processed. It may also be called after a normal card read statement. No I/O statements may intervene between a call to PRNTB and the associated read statement.

Length: 16

Subroutines used: PRNT Z, TYPE Z

### SUBROUTINE DATRD

DATRD reads one card of data according to a format previously stored by FMTRD.

Calling sequence: CALL DATRD (FORMAT, ERROR, VAR1, N1, VARZ, NZ, ..., 0, 0)

FORMAT is an integer vector fifty words long previously named in a call to FMTRD. ERROR is an integer word. VAR1, VAR2, etc. are integer or real variables or vectors. N1, N2, etc. are integer variables or constants. Each is positive if the corresponding variable is integer, negative if real.

Upon return, the first  $N_i$  locations of each  $VAR_i$  are replaced by data. Automatic type conversion from I specification to real and from E or F specification to integers is performed. If no error is detected, ERROR is set to zero; otherwise it is set to the next column to be processed. The error is either in the specified column or in the preceding field. None of the  $N_i$  may be zero. Two zeros end the list of variables.

Data: Only one data card can be read by this routine. An attempt to read beyond the end of the format is treated as an error. Numbers

may have any number of leading or trailing blanks. Signs may have leading and trailing blanks. If the sign is omitted, it is assumed to be positive. For F and E conversions, a decimal point is allowed; if omitted it is implied by the format. E type numbers may have an exponent part which must start with an E, a blank or a sign. Blanks may not precede the E. If the exponent minus the number of decimals (explicit or implicit) is not in the range  $\pm 63$ , an error is indicated. If the absolute value of the number ignoring the decimal point and exponent is greater than  $2^{31}-1$ , the result will be incorrect with no error indication given. An overflow or underflow condition is possible and is ignored.

Length: 350

Subroutines required: CARDZ, NORM, GMPYX, GDIVX, IFIX,  
FSTOX

SUBROUTINE NAME: GMPYX

GMPYX is equivalent to EMPYX, from the IBM 1130 FORTRAN Library.

SUBROUTINE NAME: GDIVX

GDIVX is equivalent to EDIVX, from the IBM 1130 FORTRAN Library. (GMPYX and GDIVX are required by DATRD in a form accessible to assembly language routines.)

The following routine is written in FORTRAN.

SUBROUTINE NAME: FMAT

FMAT is called to allow correct output from the typewriter or printer; when a format statement is handled by the typewriter, the carriage control character is printed unless FMAT is used.

The following routine is called on user option by each of the four system programs. It is included to aid the user in preparing a program for variable transformation. The User's Manual which is distributed with the 1130 Statistical System discusses such a program.

#### SUBROUTINE TRAN

TRAN is a user written subroutine which currently returns to the calling program.

## 4.0 PROGRAMMING NOTES

An experienced system user may desire to modify sections of the package. For example, larger arrays could be analyzed by modifying dimension statements, primarily those evident in COMMON. Such revision may be desirable in Regression and Factor Analysis, and may require that the number of main linkages be increased, to provide adequate storage facilities. In Orthogonal Polynomials, if scaling is used, and the user desires the original coefficients for his polynomial, those for X, rather than X', another link could be written to provide these. Considerable care should be taken concerning accuracy, so that the same problems do not arise that were bothersome in the unscaled situation.

In Factor Analysis, if the user retains the correlation matrix by saving it on the disk throughout the calculation, then factor scores could be calculated by the direct method, rather than the short method. The short method only calls for inversion of an m-by-m matrix, where m is the number of factors rotated. Modifications to allow direct estimation will require revision of links FCTR1, FCTR2, and FCTR3.

The following table gives core requirements for each program in the 1130 Statistical System using the 1130 Disk Monitor System, Mod. Level 2.

Program	Variables	Common	Program	Total
FMTRD, PRNTB				
DATRD, GMPY				
GDIV				578
TRAN	0	0	4	4
MXRAD	14	2142	234	2390
COREL	24	2262	656	2942
PRNT	8	2142	668	2818
PCOEF	14	1540	292	1846
POLY	32	1182	1024	2238
POL2	8	3232	62	3302
POLSQ	40	3232	1042	4314
PFIT	14	2032	318	2364
PDER	24	1438	432	1894
REGR	24	2412	1378	3814
REGR2	8	2262	48	2318
REGRE	112	2262	1902	4276
ANOVA	28	3166	724	3918
STORE	2	3166	44	3212
GET	2	3166	42	3210
ANOV2	14	4166	44	4224
SDOP	16	3166	206	3388

Program	Variables	Common	Program	Total
MNSQ	10	3166	348	3524
REPRT	34	3206	690	3930
FCTR	30	2412	1782	4224
FCTR1	20	2264	252	2536
INVR5	10	0	322	332
XMAX	2	0	28	30
TRIDI	156	2264	826	3246
QR	154	2264	638	3056
FCTR2	88	2264	480	2832
RFOUT	6	1342	596	1944
PROMX	12	1362	578	1952
VARMX	76	1142	1068	2286
VECTR	136	2264	530	2930
COVEC	196	2264	322	2782
FCTR3	14	1362	66	1442
RPRNT	8	942	808	1758
MATIN	72	0	482	554
SCORE	16	1162	822	2000
FMAT	0	0	34	34

## 5.0 LIST OF SWITCHES

One console entry switch is used by the 1130 Statistical System. If switch 15 is off (down), then each time a program punches cards, a message reminds the user to supply blank cards. This reminder can be suppressed by turning switch 15 on (up).

# 6.0 PROGRAM LISTINGS

// ASM READ VARIABLE FORMAT  
\* READ AND DECODE FORMAT CARDS

```
*
* ENT FMTRD
*
LPREN DC
BSI READ
DC .(
MDX *+1
MDX *+3
BSI RFAD
DC .(
MDX *+2
MDX L LPREN,1
BSC I LPREN
*
RPREN DC
BSI READ
DC .)
MDX *+1
MDX *+3
BSI READ
DC .)
MDX *+2
MDX L RPREN,1
BSC I RPREN
*
FMTRD DC
STX 1 FMTEX-1
STX 2 FMTEX-3
STX 3 FMTEX-5
LD ZERO
LIBF CARDZ
LDX 11 FMTRD
LD 1 0
A ONE
STD STORE-2
LD 1 1
STD FMTEX-7
MDX 1 2
STX 1 FMTEX+1
LDX 1 /3C
LDX 2 -51
LD NUL
STD 1 80
*
BEGIN BSI LPREN
MDX FMTER
LD AD5
BSI STORE
BSI NUMBR
NDP
BSI LPREN
MDX ELEN-2
LD NUM
```

```
FMRD 0
FMRD 10
FMRD 20
FMRD 30
FMRD 40
FMRD 50
FMRD 60
FMRD 70
FMRD 80
FMRD 90
FMRD 100
FMRD 110
FMRD 120
FMRD 130
FMRD 140
FMRD 150
FMRD 160
FMRD 170
FMRD 180
FMRD 190
FMRD 200
FMRD 210
FMRD 220
FMRD 230
FMRD 240
FMRD 250
FMRD 260
FMRD 270
FMRD 280
FMRD 290
FMRD 300
FMRD 310
FMRD 320
FMRD 330
FMRD 340
FMRD 350
FMRD 360
FMRD 370
FMRD 380
FMRD 390
FMRD 400
FMRD 410
FMRD 420
FMRD 430
FMRD 440
FMRD 450
FMRD 460
FMRD 470
FMRD 480
FMRD 490
FMRD 500
FMRD 510
FMRD 520
FMRD 530
FMRD 540
```

```
EDR ONE
STO SW
BSC L REP,+-
EDR AD2
BSI STORE
STX 2 HOLD
BSI NUMBR
NDP
BSI SPECIF
MDX FMTER
BSI READ
DC .,
MDX *+1
MDX REP
BSI RPREN
MDX FMTER
LD SW
BSC L ELFN,+-
LD AD3
A HOLD
BSI STORE
MDX ELEN
BSI SPECIF
MDX FMTER
ELEN BSI READ
DC .,
MDX *+1
MDX ELFM
BSI RPREN
MDX FMTER
LD AD4
BSI STORE
LDX 1 0
STX L1
LDX L3
LDX L2 0
LDX L1 0
FMTEX BSC L 0
*
READ DC
BSI GETCL
LDX I3 READ
EDR 3
BSC L *+2,Z
MDX 1 1
MDX 3 1
BSC L3 1
*
STD L2
BSC L
STORE EQU *-1
MDX 2 1
MDX STORE-3
*
FMTER MDX 1 1-/3C
```

```
FMRD 550
FMRD 560
FMRD 570
FMRD 580
FMRD 590
FMRD 600
FMRD 610
FMRD 620
FMRD 630
FMRD 640
FMRD 650
FMRD 660
FMRD 670
FMRD 680
FMRD 690
FMRD 700
FMRD 710
FMRD 720
FMRD 730
FMRD 740
FMRD 750
FMRD 760
FMRD 770
FMRD 780
FMRD 790
FMRD 800
FMRD 810
FMRD 820
FMRD 830
FMRD 840
FMRD 850
FMRD 860
FMRD 870
FMRD 880
FMRD 890
FMRD 900
FMRD 910
FMRD 920
FMRD 930
FMRD 940
FMRD 950
FMRD 960
FMRD 970
FMRD 980
FMRD 990
FMRD1000
FMRD1010
FMRD1020
FMRD1030
FMRD1040
FMRD1050
FMRD1060
FMRD1070
FMRD1080
FMRD1090
```

```

      MDX      FMTEX-8
*
NUM  DC
AD1  DC      /0081
AD2  DC      /0101
AD3  DC      /0180+51
AD4  DC      /0200
OR0  DC      0
DP1  DC      /4000
DR2  DC      /8000
DR3  DC      /C000
DNE  DC      1
WORK DC
AD5  DC      /0281
SW   DC
HDL0 DC
ZER0 EQU     DP0
NUL  EQU     ZER0
CHZER DC      .0
BLNK DC
*
NUMBR DC
LD     ONE
STD    NUM
BS1    DIGIT
MDX    NUMEX
STD    NUM
BS1    DIGIT
MDX    NUMEX-3
LD     NUM
SLA    2
A      NUM
SLA    1
A      DIG
STD    NUM
LD     NUM
MDX    L     NUMBR,1
NUMEX  BSC   1   NUMBR
*
SPCIF DC
BS1    READ
DC      .X
MDX    **2
LD     DP0
MDX    SPCEX-4
LD     NUM
EDR    ONE
BSC    L     **2,+
EDR    AD1
BS1    STORE
BS1    READ
DC      .I
MDX    **5
BS1    NUMBR
MDX    FMTER

```

```

FMRD1100
FMRD1110
FMRD1120
FMRD1130
FMRD1140
FMRD1150
FMRD1160
FMRD1170
FMRD1180
FMRD1190
FMRD1200
FMRD1210
FMRD1220
FMRD1230
FMRD1240
FMRD1250
FMRD1260
FMRD1270
FMRD1280
FMRD1290
FMRD1300
FMRD1310
FMRD1320
FMRD1330
FMRD1340
FMRD1350
FMRD1360
FMRD1370
FMRD1380
FMRD1390
FMRD1400
FMRD1410
FMRD1420
FMRD1430
FMRD1440
FMRD1450
FMRD1460
FMRD1470
FMRD1480
FMRD1490
FMRD1500
FMRD1510
FMRD1520
FMRD1530
FMRD1540
FMRD1550
FMRD1560
FMRD1570
FMRD1580
FMRD1590
FMRD1600
FMRD1610
FMRD1620
FMRD1630
FMRD1640

```

```

// DUP
*STORE

```

```

WS UA FMTRD

```

```

SLA    7
DR      DR1
MDX    SPCEX-3
BS1    READ
DC      .F
MDX    **2
LD     DP2
MDX    **4
BS1    READ
DC      .E
MDX    SPCEX
LD     DP3
STD    WORK
BS1    NUMBR
MDX    FMTER
SLA    7
DR      WORK
STD    WORK
BSJ    READ
DC      .
MDX    FMTER
BS1    NUMBR
MDX    FMTER
LD     WORK
OR      NUM
BS1    STORE
MDX    L     SPCIF,1
SPCEX  BSC   1   SRCIF
*
GETCL  DC
LD      1 0
EDR     BLNK
BSC    L   **2,Z
MDX    1 1
MDX    GETCL+1
EDR     BLNK
BSC    1   GETCL
*
DIG     DC
DIGIT  BSS    1
BS1    GETCL
S      CHZER
BSC    L   DIGEX,+Z
STD    DIG
MDX    1 1
MDX    L   DIGIT,1
DIGEX  BSC   1   DIGIT
END

```

```

FMRD1650
FMRD1660
FMRD1670
FMRD1680
FMRD1690
FMRD1700
FMRD1710
FMRD1720
FMRD1730
FMRD1740
FMRD1750
FMRD1760
FMRD1770
FMRD1780
FMRD1790
FMRD1800
FMRD1810
FMRD1820
FMRD1830
FMRD1840
FMRD1850
FMRD1860
FMRD1870
FMRD1880
FMRD1890
FMRD1900
FMRD1910
FMRD1920
FMRD1930
FMRD1940
FMRD1950
FMRD1960
FMRD1970
FMRD1980
FMRD1990
FMRD2000
FMRD2010
FMRD2020
FMRD2030
FMRD2040
FMRD2050
FMRD2060
FMRD2070
FMRD2080
FMRD2090
FMRD2100
FMRD2110
FMRD2120
FMRD2130
FMRD2140

```

// ASM

\* RPRINT I-O BUFFER (80 CHARACTERS, DOUBLE SPACE)  
\* RPRINT I-O BUFFER (80 CHARACTERS, DOUBLE SPACE)  
\*

ENT PRNTB  
\*  
PRNTB DC  
STX 2 SVE+1  
LDX 2 80  
LD 2 /3C-1  
STO 2 /3C  
MDX 2 -1  
MDX \*-4  
LD CN1  
STO 2 /3C  
LDX 2 B1  
LIBF PRNTZ  
SVE LDX L2  
CN1 BSC I RRNTB  
DC .0  
END

// DUP  
\*STORE 03WS UA RRNTB

PRNB 0  
PRNB 10  
PRNB 20  
PRNB 30  
PRNB 40  
PRNB 50  
PRNB 60  
PRNB 70  
RRNB 80  
RRNB 90  
PRNB 100  
PRNB 110  
PRNB 120  
PRNB 130  
PRNB 140  
RRNB 150  
PRNB 160  
PRNB 170  
PRNB 180  
PRNB 190  
RRNB 200  
PRNB 210

// ASM

\* READ DATA ACCORDING TO FORMAT STATEMENT  
\* READ DATA ACCORDING TO FORMAT STATEMENT  
\*

ENT DATRD  
DBL A I2 1  
STO 3 125  
A 2 0  
STO \*\*7  
SLA 16  
S 3 125  
STO L 1  
BSI FMTEN  
MDX DATER  
LIBF ESTJX  
DC 0  
MDX 1 -2  
MDX \*-6  
MDX LIST

\*  
DATER MDX 2 2  
LD I2 1  
BSC L DATER,Z  
LD COLM+1  
S CN1

\*  
MDX 2 2  
STX 2 DATEX+1  
STO L 0  
LDX L2 0  
LDX L1 0  
DATEX BSC L 0

\*  
DATRD DC  
STX 1 DATEX-1  
STX 2 DATEX-3  
SRA 16  
STO SPEC+1  
LDX 1 /3C  
STX 1 COLM+1  
STO 1 80  
LIBF CARDZ  
LDX I2 DATRD  
LD 2 0  
STO SPEC+3  
MDX L SPEC+3,-49  
LD 2 1  
STO DATEX-5

\*  
LIST MDX 2 2  
LD I2 1  
BSC L DATEX-3,+  
BSC L DBL,+  
STO L 1  
LD 2 0  
S L 1

DTRD 0  
DTRD 10  
DTRD 20  
DTRD 30  
DTRD 40  
DTRD 50  
DTRD 60  
DTRD 70  
DTRD 80  
DTRD 90  
DTRD 100  
DTRD 110  
DTRD 120  
DTRD 130  
DTRD 140  
DTRD 150  
DTRD 160  
DTRD 170  
DTRD 180  
DTRD 190  
DTRD 200  
DTRD 210  
DTRD 220  
DTRD 230  
DTRD 240  
DTRD 250  
DTRD 260  
DTRD 270  
DTRD 280  
DTRD 290  
DTRD 300  
DTRD 310  
DTRD 320  
DTRD 330  
DTRD 340  
DTRD 350  
DTRD 360  
DTRD 370  
DTRD 380  
DTRD 390  
DTRD 400  
DTRD 410  
DTRD 420  
DTRD 430  
DTRD 440  
DTRD 450  
DTRD 460  
DTRD 470  
DTRD 480  
DTRD 490  
DTRD 500  
DTRD 510  
DTRD 520  
DTRD 530  
DTRD 540

```

STD      **4
BSI      FMTEN
MDX      DATER
LIBF     IFIX
STD      L1 0
MDX      L  -1
MDX      *-7
MDX      LIST
*
CN1      DC      /3C-1
XR1      DC
XR2      DC
*
MISC     LDX      L1 BRTB+4
          SLA      9
          SLT      7
          BSC      11
WIDTH    EQU      *-1
XTYPE    A        COLM+1
          STD      COLM+1
          MDX      SPEC
          AXT1     STD      XR1
          MDX      SPEC
          AXT2     STD      XR2
          MDX      SPEC
          TIX2     MDX      L XR2,-1
          STD      SPEC+1
          MDX      SPEC
          INIT     EQU      AXT1
          FMTEN    DC
          STX      1 FMTX+1
          STX      2 FMTX+3
          SPEC     LDX      L1
          LD        L1 0
          MDX      L  XR1,-1
          MDX      **3
          MDX      L  XR1,1
          MDX      1 1
          STX      1 SPEC+1
          SRT      14
          STD      L 1
          SLA      9
          SLT      7
          STD      WIDTH
          BSC      11 BRTB+2
*
GETCL    DC
          LD        WIDTH
          BSC      L  **4,+
          LD        L
          MDX      L  GETCL,1
          BSC      1  GETCL
*
BLNKS    DC
          BSI      GETCL

```

```

DTRD 550
DTRD 560
DTRD 570
DTRD 580
DTRD 590
DTRD 600
DTRD 610
DTRD 620
DTRD 630
DTRD 640
DTRD 650
DTRD 660
DTRD 670
DTRD 680
DTRD 690
DTRD 700
DTRD 710
DTRD 720
DTRD 730
DTRD 740
DTRD 750
DTRD 760
DTRD 770
DTRD 780
DTRD 790
DTRD 800
DTRD 810
DTRD 820
DTRD 830
DTRD 840
DTRD 850
DTRD 860
DTRD 870
DTRD 880
DTRD 890
DTRD 900
DTRD 910
DTRD 920
DTRD 930
DTRD 940
DTRD 950
DTRD 960
DTRD 970
DTRD 980
DTRD 990
DTRD1000
DTRD1010
DTRD1020
DTRD1030
DTRD1040
DTRD1050
DTRD1060
DTRD1070
DTRD1080
DTRD1090

```

```

          MDX      **5
          EDR      BLNK
          BSC      1 BLNKS,Z
          BSI      STPCL
          MDX      BLNKS+1
          MDX      L  BLNKS,1
          MDX      BLNKS+3
          BLNK     DC
*
STPCL    DC
          MDX      L  CDLM+1,1
          MDX      L  WIDTH,-1
          NDP
          BSC      1  STPCL
*
CHZER    DC      .0
NUMEX    DC
          BSI      GETCL
          MDX      NUMXX
          S        CHZER
          BSC      L  NUMXX,+Z
          STD      DIG+1
          BSI      STPCL
          CNTSW    MDX      L  COUNT,0
          LDD      NUM
          SLT      2
          AD        NUM
          SLT      1
          SGN      AD      DIG
          STD      NUM
          MDX      NUMEX+1
          NUMXX    BSC      1  NUMEX
*
          STD      COUNT
          DP        DC
          DC        TABLE+1B
          LD        COUNT
          MDX      SCL+2
          SCALE    LD      EDIVX
          STD      OP
          LD        COUNT
          BSC      L  **4,-
          LD        EMPYX
          STD      OP
          LD        ZERO
          S        COUNT
          SCL      LDX      1 -1B
          BSC      L  OP-1,E
          SRA      1
          MDX      1 3
          MDX      SCL
          BSC      +
          CMMN     BSI      BLNKS
          MDX      FMTX
          MDX      L  FMTEN,1

```

```

DTRD1100
DTRD1110
DTRD1120
DTRD1130
DTRD1140
DTRD1150
DTRD1160
DTRD1170
DTRD1180
DTRD1190
DTRD1200
DTRD1210
DTRD1220
DTRD1230
DTRD1240
DTRD1250
DTRD1260
DTRD1270
DTRD1280
DTRD1290
DTRD1300
DTRD1310
DTRD1320
DTRD1330
DTRD1340
DTRD1350
DTRD1360
DTRD1370
DTRD1380
DTRD1390
DTRD1400
DTRD1410
DTRD1420
DTRD1430
DTRD1440
DTRD1450
DTRD1460
DTRD1470
DTRD1480
DTRD1490
DTRD1500
DTRD1510
DTRD1520
DTRD1530
DTRD1540
DTRD1550
DTRD1560
DTRD1570
DTRD1580
DTRD1590
DTRD1600
DTRD1610
DTRD1620
DTRD1630
DTRD1640

```

```

FMTEX LDX L1 0
      LDX L2 0
      BSC I FMTEX
HLT EQU FMTEX
*
DIG DEC 0
ZERO DEC 0
*
READ DC
      LDX I2 READ
      BSI GETCL
      MDX **5
      EOR 2 0
      BSC L **2,Z
      BSI STPCL
      MDX 2 1
      BSC L2 1
*
NUM DEC 0
      MDX L NUMBR,1
      BSC L
NUMBR EQU *-1
      BSI BLNKS
      NOP
      LDD ZERO
      STD NUM
      LD SWOFF
      STD CNTSW
      BSI NUMEX
      BSI READ
      DC **
      MDX NUMBR-3
      STD COUNT
      LD SWON
      STD CNTSW
      BSI NUMEX
      MDX NUMBR-1
*
EMPYX LIBF GMPYX
EDIVX LIBF GDIVX
COUNT DC
SWON DC /7401 MDX L ,1
ISWOF NOP
ISWON MDX X FMTEX-ISW-1
EXP DC 159
SWOFF DC /4C3B BSC L ,+Z-
*
ETYPE LD ISWOF
      BSI FFI
      BSI READ
      DC .E
      MDX **3
      BSI SIGN
      NOP
      MDX **2

```

```

DTRD1650
DTRD1660
DTRD1670
DTRD1680
DTRD1690
DTRD1700
DTRD1710
DTRD1720
DTRD1730
DTRD1740
DTRD1750
DTRD1760
DTRD1770
DTRD1780
DTRD1790
DTRD1800
DTRD1810
DTRD1820
DTRD1830
DTRD1840
DTRD1850
DTRD1860
DTRD1870
DTRD1880
DTRD1890
DTRD1900
DTRD1910
DTRD1920
DTRD1930
DTRD1940
DTRD1950
DTRD1960
DTRD1970
DTRD1980
DTRD1990
DTRD2000
DTRD2010
DTRD2020
DTRD2030
DTRD2040
DTRD2050
DTRD2060
DTRD2070
DTRD2080
DTRD2090
DTRD2100
DTRD2110
DTRD2120
DTRD2130
DTRD2140
DTRD2150
DTRD2160
DTRD2170
DTRD2180
DTRD2190

```

```

      BSI SIGN
      MDX SCALE
      BSI NUMBR
      MDX FMTEX
      LD COUNT
      S NUM+1
      STD COUNT
      MDX SCALE
*
ITYPE LD ISWON
      MDX FTYPE+1
FTYPE LD ISWOF
      BSI FFI
      MDX SCALE
*
PLUS AD X DIG-SGN-1
MINUS SD X DIG-SGN-1
      LD MINUS
      STD SGN
      LDX 1 1
      BSC L1 0
SIGN EQU *-1
      LD PLUS
      STD SGN
      LDX 1 0
      BSI READ
      DC .
      MDX **2
      LDX 1 1
      MDX *-5
      BSI READ
      DC .+
      MDX **1
      MDX SIGN-2
      BSI READ
      DC .+
      MDX **1
      MDX SIGN-2
      BSI READ
      DC .+
      MDX **1
      MDX SIGN-1
      MDX SIGN-4
*
FFIX DC
      STD ISW
      SLA 9
      SLT 7
      STD COUNT
      BSI SIGN
      NOP
      BSI NUMBR
      MDX FMTEX
      LDD NUM
      STD 3 126
      LD EXP

```

```

DTRD2200
DTRD2210
DTRD2220
DTRD2230
DTRD2240
DTRD2250
DTRD2260
DTRD2270
DTRD2280
DTRD2290
DTRD2300
DTRD2310
DTRD2320
DTRD2330
DTRD2340
DTRD2350
DTRD2360
DTRD2370
DTRD2380
DTRD2390
DTRD2400
DTRD2410
DTRD2420
DTRD2430
DTRD2440
DTRD2450
DTRD2460
DTRD2470
DTRD2480
DTRD2490
DTRD2500
DTRD2510
DTRD2520
DTRD2530
DTRD2540
DTRD2550
DTRD2560
DTRD2570
DTRD2580
DTRD2590
DTRD2600
DTRD2610
DTRD2620
DTRD2630
DTRD2640
DTRD2650
DTRD2660
DTRD2670
DTRD2680
DTRD2690
DTRD2700
DTRD2710
DTRD2720
DTRD2730
DTRD2740

```

```
// DUP
*STORE      WS  UA  DATRD
```

```

      STO 3 125
      LIBF NORM
FFIXX BSC I FFI
*
BRTB DC FTYPE
      DC ETYPE
      DC MISC
      DC ITYPE
      DC XTYPE
      DC AXT1
      DC AXT2
      DC TIX2
      DC HLT
      DC INIT
TABLE DC /0084      10.E01,TRUNCATED
      DC /5000
      DC /0000
      DC /0087      10.E02,TRUNCATED
      DC /6400
      DC /0000
      DC /009E      10.E04,TRUNCATED
      DC /4E20
      DC /0000
      DC /0098      10.E08,TRUNCATED
      DC /5F5E
      DC /1000
      DC /0086      10.E16,ROUNDED
      DC /470D
      DC /E4E0
      DC /00E8      10.E32,TRUNCATED
      DC /4EE2
      DC /D6D4
      END
```

```

DTRD275C // ASM
DTRD2760
DTRD2770
DTRD2780
DTRD2790
DTRD2800
DTRD2810
DTRD2820
DTRD2830
DTRD2840
DTRD2850
DTRD2860
DTRD2870
DTRD2880
DTRD2890
DTRD2900
DTRD2910
DTRD2920
DTRD2930
DTRD2940
DTRD2950
DTRD2960
DTRD2970
DTRD2980
DTRD2990
DTRD3000
DTRD3010
DTRD3020
DTRD3030
DTRD3040
DTRD3050
DTRD3060
DTRD3070
DTRD3080
DTRD3090
```

```
// DUP
*STORE      02WS UA GMPYX
```

```

*EMPTY/EMPTYX--EXTENDED PRECISION FLOAT MULTIPLY GMPY 0
*EMPTY/EMPTYX--EXTENDED PRECISION FLOAT MULTIPLY GMPY 10
      LIBR GMPY 20
      ENT GMPYX GMPY 30
EMPTYX STX 1 EMX1+1 SAVE XR1 GMPY 40
GMPYX EQU EMPYX GMPY 50
      LD L *-* LOADER INSERT. GMPY 60
      EMC STO **3 GMPY 70
      A MCN+1 =1 SET UP EXIT. GMPY 80
      STO MX+1 GMPY 90
      MDX 11 *-* OPND ADDRESS INTO XR1. GMPY 100
      CMN LD 3 125 COMPUTE PRODUCT EXPONENT. GMPY 110
      A 1 0 GMPY 120
      S MCN =128 GMPY 130
      STO 3 125 GMPY 140
      LD 1 2 PICK UP ARG FRACTION. GMPY 150
      RTE 16 GMPY 160
      LD 1 1 GMPY 170
      LIBF XMD MULTIPLY FRACTIONS. GMPY 180
      STO 3 126 GMPY 190
      BSC +- GMPY 200
      STO 3 125 GMPY 210
      SLT 1 GMPY 220
      EOR 3 126 GMPY 230
      BSC L **5,+ GMPY 240
      EOR 3 126 GMPY 250
      STO 3 126 GMPY 260
      LD 3 125 GMPY 270
      S MCN+1 =1 GMPY 280
      STO 3 125 GMPY 290
      EMX1 LDX L1 RESTORE XR1. GMPY 300
      LIBF FARC GMPY 310
      MX BSC L *-* EXIT. GMPY 320
      MCN DC 128 0 GMPY 330
      DC 1 1 GMPY 340
      END GMPY 350
      GMPY 360
      GMPY 370
      GMPY 380
```

// ASM

```

*EDIV/EDIVX--EXTENDED PRECISION FLOAT DIVIDE
*EDIV/EDIVX--EXTENDED PRECISION FLOAT DIVIDE
LIBR
ENT      GDIVX
EDIVX STX 1 EDX1+1  SAVE XR1
GDIVX EQU EDIVX
LD      L  *- *    LOADER INSERT.
EDC     STO  **3
A       ONE+1      =1  SET UP EXIT.
STO     EDX1+3
MDX     I1 *- *    DPND ADDRESS INTD XR1.
NDP
LD      1 2
RTE     16
LD      1 1
BSC     L DDVL,+ - CHECK X/D.
STD     DVR
LD      3 126
BSC     L EDX1,+ - DIVIDEND ZERD TEST.
EOR     1 1
AND     EDCN       =/BDDD
STD     QSGN       SIGN OF QUDTIENT.
BSC     L **3,+Z
LDD     3 126      SUBTRACT MAG. OF DIVISOR
SD      DVR        FROM DIVIDEND MAGNITUDE,
MDX     **2        TO ENSURE DIVIDEND SMALLER
LDD     3 126      THAN DIVISDR.
AD      DVR
STD     3 126
DR      3 127
BSC     L **3,Z
LDD     DF1
OR      QSGN
MDX     X
LDD     DVR
LIBF    XDD
EOR     EDCN       =/BDDD
STD     3 126
EDR     QSGN
BSC     L **9,-
EOR     QSGN
BSC     -
AD      ONE
SRT     1
EDR     EDCN       =/BDDD
X      STD 3 126
LD      3 125
A       ONE+1
STD     3 125
LD      3 125      CDMPUTE QUOTIENT EXPDNENT.
S       1 0
A       EDCN+1     =128
DVL     STD 3 125
LIBF    FARC
EDX1    LDX L1 *- *  RESTORE XR1.

```

GDIV 0  
GDIV 10  
GDIV 20  
GDIV 30  
GDIV 40  
GDIV 50  
GDIV 60  
GDIV 70  
GDIV 80  
GDIV 90  
GDIV 100  
GDIV 110  
GDIV 120  
GDIV 130  
GDIV 140  
GDIV 150  
GDIV 160  
GDIV 170  
GDIV 180  
GDIV 190  
GDIV 200  
GDIV 210  
GDIV 220  
GDIV 230  
GDIV 240  
GDIV 250  
GDIV 260  
GDIV 270  
GDIV 280  
GDIV 290  
GDIV 300  
GDIV 310  
GDIV 320  
GDIV 330  
GDIV 340  
GDIV 350  
GDIV 360  
GDIV 370  
GDIV 380  
GDIV 390  
GDIV 400  
GDIV 410  
GDIV 420  
GDIV 430  
GDIV 440  
GDIV 450  
GDIV 460  
GDIV 470  
GDIV 480  
GDIV 490  
GDIV 500  
GDIV 510  
GDIV 520  
GDIV 530  
GDIV 540

// DUP  
\*STDRE

D2WS UA GDIVX

```

BSC L *- *
LD   ONE+1
STD  3 123
MDX  EDX1
QSGN DC 0
ONE  DEC 1
DVR  DEC 0
DF1  DEC 1.DB1
EDCN DC /BDDD
DC   128
END

```

```

EXIT.
TURN ON PRDGRAM DIVIDE
CHECK INDICATOR.
DIVISOR BUFFER.
0
1

```

GDIV 550  
GDIV 560  
GDIV 570  
GDIV 580  
GDIV 590  
GDIV 600  
GDIV 610  
GDIV 620  
GDIV 630  
GDIV 640  
GDIV 650  
GDIV 660  
GDIV 670

```

// FOR DUMMY SUBROUTINE FOR TRANSFORMATIONS
*ONE WORD INTEGERS
C   DUMMY SUBROUTINE FOR TRANSFORMATIONS
    SUBROUTINE TRAN
    RETURN
    END
// DUP
*STORE      WS  UA  TRAN

```

TRAN 0	// FOR SUBROUTINE TO READ AND ADD MATRICES	MXRD 0
TRAN 10	*ONE WORD INTEGERS	MXRD 10
TRAN 20	C SUBROUTINE TO READ AND ADD MATRICES	MXRD 20
TRAN 30	SUBROUTINE MXRAD	MXRD 30
TRAN 40	COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,IPRED,	MXRD 40
TRAN 50	1 ISTEP,ICNST,IREAR,KXI(1),MX(20),NC01,NC02,NC03,ISEQ,NCASF,NX(10),	MXRD 50
TRAN 60	2 EFOUT,EFIN,TOL,FLVB(2),KNN	MXRD 60
TRAN 70	COMMON TITLE(18),VNAME(30),SUMY(30),SD(30),X(30),R(30,30)	MXRD 70
	101 FORMAT(I4,3I2,5E14.7)	MXRD 80
	IKT=0	MXRD 90
	9 READ(ICR,101) IP,MIO,IC,IR,(X(I),I=1,5)	MXRD 100
	IF(IKT)51,52,51	MXRD 110
	51 DO 53 I=1,5	MXRD 120
	53 X(I)=-X(I)	MXRD 130
	52 IF(IP) 30,30,10	MXRD 140
	10 IF(MIO-2I) 11,15,20	MXRD 150
	C STORE MATRIX	MXRD 160
	11 J1 = IC	MXRD 170
	J2 = IC+4	MXRD 180
	IF (J2-N) 14,14,13	MXRD 190
	13 J2=N	MXRD 200
	14 K=0	MXRD 210
	MXT = MIO	MXRD 220
	DO 12 J=J1,J2	MXRD 230
	K = K+1	MXRD 240
	12 R(IR,J) = R(IR,J) + X(K)	MXRD 250
	GO TO 9	MXRD 260
	C STORE NUMBER OF CASES	MXRD 270
	15 CASES = CASES + X(1)	MXRD 280
	GO TO 9	MXRD 290
	C STORE MEANS AND STANDARD DEVIATIONS	MXRD 300
	20 SUMY(IR) = SUMY(IR) + X(1)	MXRD 310
	SD(IR) = SD(IR) + X(2)	MXRD 320
	GO TO 9	MXRD 330
	30 IF(MXT-1) 31,31,32	MXRD 340
	31 NCASE=MXT	MXRD 350
	1 IF(ICNST)50,35,50	MXRD 360
	50 ICNST=0	MXRD 370
	IKT=1	MXRD 380
	GO TO 9	MXRD 390
	32 IF (MXT-4) 34,34,35	MXRD 400
	34 NCASE=-MXT	MXRD 410
	GO TO 1	MXRD 420
	35 RETURN	MXRD 430
	END	MXRD 440
	// DUP	MXRD 450
	*STORE      WS  UA  MXRAD	MXRD 460

```

// FOR SUBROUTINE TO COMPUTE CORRELATION COEFFICIENTS
*IOCS(CARD,1132 PRINTER,DISK)
*NAME COREL
*ONE WORD INTEGERS
C SUBROUTINE TO COMPUTE CORRELATION COEFFICIENTS
COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,KX(5),
IMX(20),NX(15),FLVB(5),KNN
COMMON TITLE(18),VNAME(30),SUMY(30),SD(30),DATA(30),R(30,30)
COMMON HIGH(30),HLOW(30)
DEFINE FILE 606(500,65,U,IT1)
DEFINE FILE 5(30,60,U,IT2)
103 FORMAT( 10X,18A4,5X,3HJOB,17,5X,4HPAGE,I6)
104 FORMAT(/11X18HSUMMARY STATISTICS10X12HNO. OF CASES=,16, //16X8HVAR1
1ABLE16X3HLOW9X4HHIGH9X7HAVERAGE7X9HSTD. DEV.6X8HVARIANCE//)
105 FORMAT(16X,12,2X,A4,5X,5E15.5)
106 FORMAT(/1X,24HTHE VARIANCE OF VARIABLE,1XA4,1X7HIS ZERO )
107 FORMAT(14,3I2,5E14.7)
108 FORMAT(1H )
C SUM OF SQUARES TAKEN FROM X-PROD MATRIX
1 ICA=CASES
ISW = 1
KON = 22
KON1 = 1
DO 10 I=1,N
10 SD(I) = R(I,I)
CALL PRNT(1,1,N,N)
IF(MX(1)-2) 15,91,91
C COMPUTE RESIDUAL X-PROD MATRIX
15 DO 20 I=1,N
DO 20 J=I,N
R(I,J) = R(I,J) - SUMY(I)*SUMY(J)/CASES
20 R(J,I) = R(I,J)
CALL PRNT(2,0,N,N)
C COMPUTE COVARIANCE MATRIX
DO 30 I=1,N
DO 30 J=I,N
R(I,J) = R(I,J)/(CASES-1.)
30 R(J,I) = R(I,J)
CALL PRNT(3,0,N,N)
C OUTPUT MEANS AND STANDARD DEVIATIONS
ISW = 2
KON = 23
NPAGE = NPAGE + 1
CALL FMT(IPR,ITW)
IF(IPR) 31,31,32
31 WRITE(ITW,103)TITLE,IPROB,NPAGE
32 WRITE(ITW,104)ICA
DO 40 I=1,N
SUMY(I) = SUMY(I)/CASES
SD(I) = SQRT(R(I,I))
40 WRITE(ITW,105) 1,VNAME(I),HLOW(I),HIGH(I),SUMY(I),SD(I),R(I,I)
C COMPUTE CORRELATION MATRIX
45 DO 90 I=1,N
IF(SD(I)) 50,50,60
50 WRITE(ITW,106) VNAME(I)
CORL 0
CORL 10
CORL 20
CORL 30
CORL 40
CORL 50
CORL 60
CORL 70
CORL 80
CORL 90
CORL 100
CORL 110
CORL 120
CORL 130
CORL 140
CORL 150
CORL 160
CORL 170
CORL 180
CORL 190
CORL 200
CORL 210
CORL 220
CORL 230
CORL 240
CORL 250
CORL 260
CORL 270
CORL 280
CORL 290
CORL 300
CORL 310
CORL 320
CORL 330
CORL 340
CORL 350
CORL 360
CORL 370
CORL 380
CORL 390
CORL 400
CORL 410
CORL 420
CORL 430
CORL 440
CORL 450
CORL 460
CORL 470
CORL 480
CORL 490
CORL 500
CORL 510
CORL 520
CORL 530
CORL 540
60 DO 90 J=1,N
IF(SD(J)) 70,70,80
70 R(I,J) = 0.0
GO TO 90
80 R(I,J) = R(I,J)/(SD(I)*SD(J))
90 R(J,I) = R(I,J)
CALL PRNT(4,0,N,N)
IF(MX(4) - 2) 95,91,91
C PUNCH MEANS, STANDARD DEV. AND NO. OF CASES
91 READ(ICR,108)
DO 92 I=1,N
92 WRITE(ICP,107) IPROB,KON,KON1,1,SUMY(I),SD(I)
KON = 21
WRITE(ICP,107) IPROB,KON,KON1,KON1,CASES
GO TO (15,95),ISW
95 IF(KNN)150,150,151
150 CALL LINK(REGR2)
151 CALL LINK(FCTR1)
END
// DUP
*STORE WS UA COREL
CORL 550
CORL 560
CORL 570
CORL 580
CORL 590
CORL 600
CORL 610
CORL 620
CORL 630
CORL 640
CORL 650
CORL 660
CORL 670
CORL 680
CORL 690
CORL 700
CORL 710
CORL 720
CORL 730
CORL 740
CORL 750

```

```

// FOR MATRIX PRINT/PUNCH ROUTINE
*ONE WORD INTEGERS
C   MATRIX PRINT/PUNCH ROUTINE
    SUBROUTINE PRNT(MID,KDDE,NR,NC)
      COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,KX(5),
      1MX(20),NX(15),FLVB(5),KNN
      COMMON TITLE(18),VNAME(30),SUMY(30),SD(30),DATA(30),R(30,30)
101  FORMAT(5X4,4X8E13.5)
102  FORMAT(5X4,4X8E13.5)
103  FORMAT( 10X18A4,5X3HJOB17, 5X,4HPAGE I6)
104  FORMAT(I4,3I2,5E14.7)
105  FORMAT(/103H READY THE PUNCH WITH BLANK CARDS AND PRESS START ON T
      THE PUNCH AND CONSOLE. TURN CONSOLE SWITCH 15 ON.)
106  FORMAT(1H )
201  FORMAT(///2X,8HVARIALE,07X,8(A4,9X)///)
202  FORMAT(3X8HVARIALE7X8(I4,8X)///)
321  FORMAT(/ 46X,28HMATRIX OF RAW CRDSS-PRDDUCTS )
322  FORMAT(/ 43X,33HMATRIX OF RESIDUAL CRDSS-PRDDUCTS )
323  FORMAT(/45X28HVARIANCE - CDVARIANCE MATRIX )
324  FORMAT(/ 42X,34HMATRIX OF CDRRELATION CDEFFICIENTS )
325  FORMAT(/45X32HMATRIX OF CHARACTERISTIC VECTORS/)
326  FORMAT(/41X36HNDORMALIZED UNRDATED FACTOR LOADINGS)
      KNME=1
      IF(MX(MID)-1)1000,1,100
        1 I = 1
        II = 8
        9 IF(NC-II) 10,11,11
        10 II = NC
        11 NPAGE = NPAGE + 1
        CALL FMAT(IPR,ITW)
        16 IF(IPR) 12,12,13
12  WRITE(ITW,103)TITLE,IPROB,NPAGE
13  GO TO (21,22,23,24,25,26),MID
21  WRITE(ITW,321)
    GO TO 30
22  WRITE(ITW,322)
    GO TO 30
23  WRITE(ITW,323)
    GO TO 30
24  WRITE(ITW,324)
    GO TO 30
25  WRITE(ITW,325)
    GO TO 30
26  WRITE(ITW,326)
30  IF(MID-5)40,41,40
40  IF(MID-6)42,41,42
41  KNME=0
    WRITE(ITW,202)(J,J=I,II)
    GO TO 43
42  WRITE(ITW,201)(VNAME(J),J=I,II)
43  DD 35 K=1,NR
    IF(KDDE) 34,33,34
33  IF(KNME)44,45,44
44  KNME=VNAME(K)
45  WRITE(ITW,101) VNAME(K),(R(K,J),J=I,II)
      PRNT 0
      PRNT 10
      PRNT 20
      PRNT 30
      PRNT 40
      PRNT 50
      PRNT 60
      PRNT 70
      PRNT 80
      PRNT 90
      PRNT 100
      PRNT 110
      PRNT 120
      PRNT 130
      PRNT 140
      PRNT 150
      PRNT 160
      PRNT 170
      PRNT 180
      PRNT 190
      PRNT 200
      PRNT 210
      PRNT 220
      PRNT 230
      PRNT 240
      PRNT 250
      PRNT 260
      PRNT 270
      PRNT 280
      PRNT 290
      PRNT 300
      PRNT 310
      PRNT 320
      PRNT 330
      PRNT 340
      PRNT 350
      PRNT 360
      PRNT 370
      PRNT 380
      PRNT 390
      PRNT 400
      PRNT 410
      PRNT 420
      PRNT 430
      PRNT 440
      PRNT 450
      PRNT 460
      PRNT 470
      PRNT 480
      PRNT 490
      PRNT 500
      PRNT 510
      PRNT 520
      PRNT 530
      PRNT 540
      GD TO 35
34  WRITE(ITW,102) VNAME(K),(R(K,J),J=I,II)
35  CONTINUE
    IF(NC-II) 36,1000,36
36  I = I+8
    II = II + 8
    GO TO 9
C   PUNCH ROUTINE
100  I = 1
    II = 5
    READ(ICR,106)
    CALL DATSW(15,JIG)
    IF(JIG-2)151,3,3
      3 WRITE(ITW,105)
      PAUSE
151  IF(NC-II) 152,153,153
152  II = NC
153  DD 154 K = 1,NR
154  WRITE(ICP,104)IPROB,MID,I ,K,(R(K,J),J=I,II)
    IF(NC-II) 155,156,155
155  I = I + 5
    II = II + 5
    GO TO 151
156  IF(MX(MID)-2) 1000,1,1000
1000 RETURN
      END
// DUP
*STORE WS UA PRNT
      PRNT 550
      PRNT 560
      PRNT 570
      PRNT 580
      PRNT 590
      PRNT 600
      PRNT 610
      PRNT 620
      PRNT 630
      PRNT 640
      PRNT 650
      PRNT 660
      PRNT 670
      PRNT 680
      PRNT 690
      PRNT 700
      PRNT 710
      PRNT 720
      PRNT 730
      PRNT 740
      PRNT 750
      PRNT 760
      PRNT 770
      PRNT 780
      PRNT 790
      PRNT 800
      PRNT 810
      PRNT 820

```

```

// FOR SUBROUTINE TO COMPUTE COEFFICIENTS OF POLYNOMIAL
* ONE WORD INTEGERS
C SUBROUTINE TO COMPUTE COEFFICIENTS OF POLYNOMIAL
  SUBROUTINE PCOEF
    COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,ISCR,
    INCASE,ICOF,IDER,NDER,IALP,INMD2,KX(5),EPS,FLVB(4),X8,X14
    COMMON TITLE(18),ID(150),X(150),Y(150),C(51),ALPHA(51),BETA(51)
    COMMON A(51),TEMP1(51),TEMP2(51),TEMP3(51)
101  FORMAT(10X18A4,5X3HJOB17,5X,4HPAGE,I6)
102  FORMAT(20X15,E20.7)
103  FORMAT(//20X,33HCOEFFICIENTS OF FITTED POLYNOMIAL//)
C PROGRAM INITIALIZATION
  B = 0.0
  KKD = NF+1
  DO 1 NN = 1,KKD
    A(NN) = C(NN)
    TEMP1(NN) = 0.0
    TEMP2(NN) = 0.0
1  TEMP3(NN) = 0.0
C BEGIN COMPUTATION
  DO 6 II = 2,KKD
    TEMP2(II) = 1.0
    DO 3 NN = 2,II
      TEMP3(NN) = TEMP2(NN-1) - TEMP2(NN)*ALPHA(II-1) - B*TEMP1(NN)
C COMPUTATION OF A COEFFICIENT
3  A(NN-1) = A(NN-1) + C(II)*TEMP3(NN)
  IF(II-KKD) 4,B,B
C RESETTING THE VECTORS FOR THE NEXT COEFFICIENT
4  DO 5 NN = 1,II
    TEMP1(NN) = TEMP2(NN)
    TEMP2(NN) = TEMP3(NN)
5  B = BETA(II-1)
C OUTPUT POLYNOMIAL COEFFICIENTS
  B NPAGE = NPAGE + 1
  CALL FMAT(IPR,ITW)
  IF(IPR) 81,81,82
81  WRITE(ITW,101) TITLE,IPROB,NPAGE
82  WRITE(ITW,103)
  DO 9 J = 1,KKD
    L = J-1
    9  WRITE(ITW,102) L,A(J)
20  RETURN
  END
// DUP
*STORE WS UA PCOEF

```

```

PCOF 0
PCOF 10
PCOF 20
PCOF 30
PCOF 40
PCOF 50
PCOF 60
PCOF 70
PCOF 80
PCOF 90
PCOF 100
PCOF 110
PCOF 120
PCOF 130
PCOF 140
PCOF 150
PCOF 160
PCOF 170
PCOF 180
PCOF 190
PCOF 200
PCOF 210
PCOF 220
PCOF 230
PCOF 240
PCOF 250
PCOF 260
PCOF 270
PCOF 280
PCOF 290
PCOF 300
PCOF 310
PCOF 320
PCOF 330
PCOF 340
PCOF 350
PCOF 360
PCOF 370
PCOF 380
PCOF 390
PCOF 400
PCOF 410
PCOF 420
PCOF 430
PCOF 440

```

```

// FOR SUBROUTINE TO INPUT DATA FOR ORTHOGONAL POLYNOMIALS
* ONE WORD INTEGERS
*IDCS(CARD,1132PRINTER,DISK)
*NAME POLY
C SUBROUTINE TO INPUT DATA FOR ORTHOGONAL POLYNOMIALS
  COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,ISCR,
  INCASE,ICOF,IDER,NDER,IALP,INMD2,KX(5),EPS,FLVB(4),X8,X14
  COMMON TITLE(18),ID(150),X(150),Y(150),C(51),ALPHA(51),BETA(51)
  COMMON MF1(50)
  DEFINE FILE 606(150,B,U,IT1)
101  FORMAT(6I2)
102  FORMAT(14,4X,18A4)
103  FORMAT(BI2,F10.4,3I2)
104  FORMAT(///43H THE X VALUES HAVE BEEN TRANSFORMED TO X'=(,E14.7,7H)
  1*X + (,E14.7,2H).)
105  FORMAT(10X,18A4,5X,3HJOB,17,5X,4HPAGE,I6//11X,2BHMAXIMUM DEGREE
  1E OF POLYNOMIAL,5X,12/11X,10HINPUT TYPE,23X,12/11X,23HPOLYNOMIAL
  30EFFICIENTS,10X,12/11X,19HCOMPUTE DERIVATIVES,14X,12/11X,19HORDER
  40F DERIVATIVE,14X,12/11X,16HPREDICTED VALUES,17X,12/11X,22HPUNCH
  50LUTION VECTORS,11X,12/11X,20HSECONDARY INPUT TYPE,13X,12/11X,18HVAPOLY
  6RIANCE CRITERION,2X,F15.9/11X,21HTRANSFORMATION SWITCH,12X,12)
106  FORMAT(//11X,7HSCALING26X,12/11X,24HIGNORE POLYNOMIAL OUTPUT9X,12)
107  FORMAT(///,* AN ILLEGAL CHARACTER HAS BEEN ENCOUNTERED IN COLUMN
  1*,13,* OF THE ABOVE FORMAT CARD.*/' CHANGE CARD AND RFRUN JOB.)*
108  FORMAT(///,* AN ILLEGAL CHARACTER HAS BEEN ENCOUNTERED AT APPROXIMATELY
  1TELY COLUMN*,13* OF THE ABOVE DATA CARD.*/' CHANGE OR REMOVE CARD
  2AND PRESS START TO CONTINUE*)
109  FORMAT(8X12,3E14.7)
110  FORMAT(///' INVALID INPUT OPTION-JOB TERMINATED ')
111  FORMAT(2E14.7,I2)
112  FORMAT(//27H X = X' (NO TRANSFORMATION))
C SUBROUTINE TO READ AND PRINT PARAMETER CARDS (POLYNOMIAL)
  KWS=1
  NPAGE = 0
  READ(2,101) ICR,ICP,IPR,ITW,IT1,IT2
  IF(IPR) 701,702,701
702  ITW=3
  GO TO 703
701  ITW=1
703  READ(ICR,102) IPROB,TITLE
  READ(ICR,103) N,INMD,ICOF,IDER,NDER,ISCR,IALP,INMD2,EPS,KX(3)
  1,KX(4),KX(5)
  CALL FMAT(IPR,ITW)
  WRITE(ITW,105) TITLE,IPROB,NPAGE,N,INMD,ICOF,IDER,NDER,ISCR,IALP,
  1INMD2,EPS,KX(3)
  WRITE(ITW,106)KX(4),KX(5)
  IF(INMD - 2) 1,5,1001
1001 IF(INMD-3)1002,1,1002
1002 WRITE(ITW,110)
  CALL EXIT
1  CALL FMTRD(MF1,IRR)
  CALL PRNTB
  IF(IRR) 2,5,2
2  WRITE(ITW,107) IRR
  CALL EXIT

```

```

POLY 0
POLY 10
POLY 20
POLY 30
POLY 40
POLY 50
POLY 60
POLY 70
POLY 80
POLY 90
POLY 100
POLY 110
POLY 120
POLY 130
POLY 140
POLY 150
POLY 160
POLY 170
POLY 180
POLY 190
POLY 200
POLY 210
POLY 220
POLY 230
POLY 240
POLY 250
POLY 260
POLY 270
POLY 280
POLY 290
POLY 300
POLY 310
POLY 320
POLY 330
POLY 340
POLY 350
POLY 360
POLY 370
POLY 380
POLY 390
POLY 400
POLY 410
POLY 420
POLY 430
POLY 440
POLY 450
POLY 460
POLY 470
POLY 480
POLY 490
POLY 500
POLY 510
POLY 520
POLY 530
POLY 540

```

```

5 NF = N
10 IF(INMO-2) 11,11,30
11 DO 14 I=1,150
    IF(INMO-1) 16,16,20
C   READ DATA FROM CARD READER
16 CALL QATRO(MF1,IRR,IO(I),1,IOR,1,      X(I),-1,Y(I),-1,0,0)
    WRITE(606'I)ID(I),IOR,      X(I),Y(I)
    IF(IRR) 17,18,17
17 CALL PRNTB
    WRITE(ITW,108) IRR
    PAUSE 10
    GO TO 16
20 READ(606'I)IO(I),IOR,      XII,Y(I)
18 IF(IO(I)) 15,15,19
19 IF(IOR) 13,13,12
12 IO(I) = -IO(I)
13 IF(KX(3)) 143, 14,143
143 CALL TRAN
14 CONTINUE
15 NCASE = I-1
    IF(KX(4))35,200,35
200 WRITE(ITW,112)
    GO TO 100
35 IF(KWS) 356,355,356
356 XN=1.0E-30
    X1=1.0E+30
    DO 39 I=1,NCASE
        IF(X(I)-XN) 37,37,36
36 XN=X(I)
37 IF(X(I)-X1) 38,39,39
38 X1=X(I)
39 CONTINUE
    XB=XN-X1
    X14=4./X8
    XB=-(X1+X1+XN+XN)/XB
355 DO 40 I=1,NCASE
40 X(I)=X14*X(I)+XB
    WRITE(ITW,104)X14,X8
    GO TO 100
C   READ ALPHA,BETA,C FROM CARD READER
30 NP1 = N&1
    KWS=0
    READ(ICR,111)X14,XB,KX(4)
    DO 31 I=1,NP1
        READ(ICR,109) K,T1,T2,T3
        ALPHA(K) = T1
        BETA(K) = T2
31 C(K) = T3
        INMO = INMO2
        GO TO 11
100 CALL LINK(POL2)
    ENO
// DUP
*STORE      WS UA POLY
// FOR      SECONDARY MAIN FOR ORTHOGONAL PDYNOMIALS

```

```

PCLY 550 *ONE WORD INTEGERS
POLY 560 *IOCS(CARO,1132PRINTER,DISK)
POLY 570 *NAME POL2
POLY 580 C SECONDARY MAIN FOR ORTHOGONAL POLYNOMIALS
POLY 590 COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMO,ISCR,
POLY 600 INCASE,ICOF,IOER,NOER,IALP,INMO2,KX(5),EPS,FLVB(4),XB,X14
POLY 610 COMMON TITLE(18),IO(150),X(150),Y(150),C(51),ALPHA(51),BETA(51)
POLY 620 COMMON YA(150),POLY(150),POLYO(150),POL(150,4),SSR(51)
POLY 630 DEFINE FILE 606(150,8,U,IT1)
POLY 640 100 FORMAT(/2X13HJOB COMPLETED)
POLY 650 IF(INMO2) 5,5,6
POLY 660 5 CALL POLSQ
POLY 670 6 IF(ICOF) 8,8,7
POLY 680 7 CALL PCOEF
POLY 690 8 IF(IDER) 10,10,9
POLY 700 9 CALL POER
POLY 710 10 IF(1SCR) 13,13,11
POLY 720 11 CALL PFIT
POLY 730 13 WRITE(ITW,100)
POLY 740 CALL EXIT
POLY 750 ENO
POLY 760 // DUP
POLY 770 *STORE      WS UA POL2
POLY 780
POLY 790
POLY 800
POLY 810
POLY 820
POLY 830
POLY 840
POLY 850
POLY 860
POLY 870
POLY 880
POLY 890
POLY 900
POLY 910
POLY 920
POLY 930
POLY 940
POLY 950
POLY 960
POLY 970
POLY 980
POLY 990
POLY1000
POLY1010
POLY1020
POLY1030
POLY1040
POLY1050
POLY1060
POLY1070
POLY1080
PCL2 0
PCL2 10
PCL2 20
PCL2 30
PCL2 40
PCL2 50
PCL2 60
PCL2 70
PCL2 80
PCL2 90
PCL2 100
PCL2 110
PCL2 120
PCL2 130
PCL2 140
PCL2 150
PCL2 160
PCL2 170
PCL2 180
PCL2 190
PCL2 200
PCL2 210
PCL2 220
PCL2 230

```

// FOR COMPUTATION OF ORTHOGONAL POLYNOMIALS

```
* ONE WORD INTEGERS
C COMPUTATION OF ORTHOGONAL POLYNOMIALS
SUBROUTINE POLSQ
COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPRO8,N,NF,CASES,NPAGE,INMO,ISCR,
INCASE,ICOF,IOER,NOER,(ALP,INMO2,KX(5),EPS,FLV8(4),X8,X14
COMMON TITLE(18),IO(150),X(150),Y(150),CI(51),ALPHA(51),BETA(51)
COMMON YA(150),POLY(150),POLYO(150),POL(150,4),SSR(51)
101 FORMAT(//68H MAX DEGREE OF POLYNOMIAL REACHED. VARIANCE CRITERION
IN NOT SATISFIED )
102 FORMAT( /40X,22HORTHOGONAL POLYNOMIALS/ 1X14HIDENTIFICATION7X2HX*9PLSQ 100
1X1HY11X2HY*10X4HY-Y*,I12,3I13,/)
103 FORMAT( 10X18A4,5X3HJ08I7,5X,4HPAGE,I6)
104 FORMAT(1H )
105 FORMAT(/103H READY THE PUNCH WITH BLANK CAROS AND PRESS START ON
THE PUNCH AND CONSOLE. TURN CONSOLE SWITCH 15 ON.)
106 FORMAT(1X13,I7,2X4E13.5,4F13.6)
107 FORMAT( 60X5HALPHA4E13.5)
108 FORMAT( 61X4HETA4E13.5)
109 FORMAT( 64X1HC4E13.5)
110 FORMAT(14,3I2,3E14.7)
111 FORMAT(2E14.7,I2)
112 FORMAT(1X(3,I7,2X8E13.5)
113 FORMAT(//22X20HANALYSIS OF VARIANCE//3X16HVARIATION SOURCE11X4HO.FPLSQ 221
1.6X28HSUM OF SQUARES MEAN SQUARE/)
114 FORMAT(3X6HDEGREE,I3,10H COMPONENT,I11,7X,2E14.5)
115 FORMAT(3X16HRESIDUALSDEGREE,I3,7H REGR.),I4,7X,2E14.5/)
C INITIALIZATION
MN=41
II=1
N1=1
IT = 1
ISW = 1
RO = NCASE
FN = RO
SY=0.0
OEL1 = 0.0
OO 10 I =1,NCASE
OEL1 = OEL1*Y(I)*Y(I)
SY=SY+Y(I)
POLYO(I) = 0.0
YA(I) = 0.0
10 POLY(I) = 1.0
Y8AR=SY/RO
SY=OEL1-SY*Y8AR
8=C.
C BEGIN COMPUTATION
11 S = 0.
SSR(II)=0.0
SSR(II+1)=0.0
OO 12 I=1,NCASE
12 S= S*Y(I)*POLY(I)
C COMPUTATION OF A COEFFICIENT IN THE POLYNOMIAL EQUATION.
C(II) = S/RO
C COMPUTE PREOICTED VALUES
OO 13 I=1,NCASE
13 YAI(I) = YA(I)&C(II)*POLY(I)
C DETERMINE IF VARIANCE CRITERION IS SATISFIED
OEL2 = OEL1-S*SY/RO
VAR2 = OEL2/(FN-II-1.0)
IF(II-NCASE+1)610,610,611
610 IF(II-1) 17,17,14
14 IF(A8S(VAR2-VAR1)-EPS) 15,16,16
```

PLSQ 0  
PLSQ 10  
PLSQ 20  
PLSQ 30  
PLSQ 40  
PLSQ 50  
PLSQ 60  
PLSQ 70  
PLSQ 80  
PLSQ 90  
PLSQ 100  
PLSQ 110  
PLSQ 120  
PLSQ 130  
PLSQ 140  
PLSQ 150  
PLSQ 160  
PLSQ 170  
PLSQ 180  
PLSQ 190  
PLSQ 200  
PLSQ 210  
PLSQ 220  
PLSQ 221  
PLSQ 222  
PLSQ 223  
PLSQ 224  
PLSQ 230  
PLSQ 240  
PLSQ 250  
PLSQ 251  
PLSQ 260  
PLSQ 270  
PLSQ 280  
PLSQ 290  
PLSQ 291  
PLSQ 300  
PLSQ 310  
PLSQ 320  
PLSQ 321  
PLSQ 330  
PLSQ 340  
PLSQ 350  
PLSQ 351  
PLSQ 352  
PLSQ 360  
PLSQ 370  
PLSQ 380  
PLSQ 381  
PLSQ 382  
PLSQ 390  
PLSQ 400  
PLSQ 410  
PLSQ 420  
PLSQ 430  
PLSQ 440  
PLSQ 450  
PLSQ 460  
PLSQ 470  
PLSQ 480  
PLSQ 490  
PLSQ 500  
PLSQ 510

```
15 II=II-1
NF=II-1
IT=IT-1
ISW=2
GO TO 45
16 IF(II-N) 17,17,61
C COMPUTATION OF ALPHA FOR THE POLYNOMIAL EQUATION
17 SUMXQ = 0.
OO 18 I=1,NCASE
18 SUMXQ = SUMXQ & X(I)*POLY(I)*POLY(I)
ALPHA(II) = SUMXQ/RO
C COMPUTATION OF A NEW POLYNOMIAL
OO 19 I=1,NCASE
POL(II,IT) = POLY(I)
POLY(I) = (X(II)-ALPHA(II))*POLY(I)-B*POLYO(I)
19 POLYO(I) = POL(II,IT)
OO 191 I=1,NCASE
SSR(II)=SSR(II)+(IY(II)-Y8AR)*POLY(II)
191 SSR(II+1)=SSR(II+1)+POLY(II)**2
SSR(II)=SSR(II)**2/SSR(II+1)
C COMPUTATION OF BETA FOR THE POLYNOMIAL EQUATION
R = 0.0
OO 20 I=1,NCASE
20 R = R & POLY(II)*POLY(II)
BETA(II) = R/RO
RO = R
B = BETA(II)
GO TO (21,45),(SW
C OUTPUT SECTION OF ORTHOGONAL POLYNOMIALS
21 IF(IT-4) 60,45,45
45 IX = II-1
IF(IX(15))507,506,507
506 IL = II-IT
NPAGE = NPAGE & 1
OO 52 I=1,NCASE
IF(I-MN)503,48,503
48 NPAGE=NPAGE+1
MN=MN+40
GO TO 47
503 IF(I-1)502,47,502
47 CALL FMAT(IPR,ITW)
IF(IPR) 461,461,478
461 WRITE(ITW,103) TITLE,IPRO8,NPAGE
478 WRITE(ITW,102)IJ,J=IL,IX)
502 DIF = Y(I) - YA(I)
IF(IX(14))528,529,528
529 WRITE(ITW,112) I,IO(I),X(I),Y(I),YA(I),OIF,(POL(I,J),J=1,IT)
GO TO 52
528 WRITE(ITW,106) I,IO(I),X(I),Y(I),YA(I),DIF,(POL(I,J),J=1,IT)
52 CONTINUE
IL = IL&1
WRITE(ITW,107) (ALPHA(I),I=IL,II)
WRITE(ITW,108) (BETA(I),I=IL,II)
WRITE(ITW,109) (C(II),I=1L,II)
507 IT = 0
GO TO (60,100),ISW
C CONTINUE THE NEXT ORDER POLYNOMIAL
60 OEL1 = OEL2
VAR1 = VAR2
II = II & 1
IT = IT & 1
GO TO 11
```

PLSQ 520  
PLSQ 530  
PLSQ 540  
PLSQ 550  
PLSQ 560  
PLSQ 570  
PLSQ 580  
PLSQ 590  
PLSQ 600  
PLSQ 610  
PLSQ 620  
PLSQ 630  
PLSQ 640  
PLSQ 650  
PLSQ 660  
PLSQ 670  
PLSQ 671  
PLSQ 672  
PLSQ 673  
PLSQ 674  
PLSQ 680  
PLSQ 690  
PLSQ 700  
PLSQ 710  
PLSQ 720  
PLSQ 730  
PLSQ 740  
PLSQ 750  
PLSQ 760  
PLSQ 770  
PLSQ 780  
PLSQ 790  
PLSQ 800  
PLSQ 810  
PLSQ 820  
PLSQ 830  
PLSQ 840  
PLSQ 850  
PLSQ 860  
PLSQ 870  
PLSQ 880  
PLSQ 890  
PLSQ 900  
PLSQ 910  
PLSQ 920  
PLSQ 930  
PLSQ 940  
PLSQ 950  
PLSQ 960  
PLSQ 970  
PLSQ 980  
PLSQ 990  
PLSQ1000  
PLSQ1010  
PLSQ1020  
PLSQ1030  
PLSQ1040  
PLSQ1050  
PLSQ1060  
PLSQ1070  
PLSQ1080  
PLSQ1090

```

61 WRITE(ITW,101)
   NF = N
65 ISW = 2
   GO TO 17
611 WRITE(ITW,101)
   NF=NCASE-1
   GO TO 65
C   TEST FOR PUNCHING OF ALPHA,BETA,C
100 IF(IALP) 140,140,125
125 NFPI = NF&1
   KON = 24
   KON1 = 1
   READ(ICR,104)
   CALL DATSW(15,JIG)
   IF(JIG-2)151,3,3
   3 WRITE(ITW,105)
     PAUSE
151 WRITE(ICP,111)X14,X8,KX(4)
   DO 126 I=1,NFPI
126 WRITE(ICP,110) IPROB,KON,KON1,I,ALPHA(I),BETA(I),C(I)
140 WRITE(ITW,113)
   DO 192 I=1,NF
   WRITE(ITW,114)I,N1,SSR(I),SSR(I)
   NDF=NCASE-1-I
   ADF=NDF
   SY=SY-SSR(I)
   AMY=SY/ADF
192 WRITE(ITW,115)I,NOF,SY,AMY
   RETURN
   END

// OUP
*STORE      WS UA POLSQ
// FOR SUBROUTINE TO COMPUTE POLYNOMIAL PREDICTED VALUES
*   DNE WORD INTEGERS
C   SUBROUTINE TO COMPUTE POLYNOMIAL PREDICTED VALUES
SUBROUTINE PFIT
COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMO,ISCR,
INCASE,ICOF,IOER,NDER,IALP,INMO2,KX(5),EPS,FLV8(4),X8,X14
COMMON TITLE(18),ID(150),X(150),Y(150),C(51),ALPHA(51),BETA(51)
COMMON YA(150),POLY(150),POLYO(150)
101 FORMAT( 10X18A4,5X3HJ08I7, 5X,4HPAGE 16/)
102 FORMAT(10X13,17,2X4E13.5)
103 FORMAT(/8X,14HIDENTIFICATION,9X2HX*12X1HY12X2HY*9X4HY-Y*//)
C   INITIALIZATION
KAJPI = NF&1
8=0.0
DO 1 I=1,NCASE
YA(I)=0.0
POLY(I)=1.0
1 POLYO(I)=0.0
DO 6 II =I,KAJPI
C   COMPUTE PREDICTED VALUES
DO 3 I=1,NCASE
3 YA(I)=YA(I)&C(II)*POLY(I)
IF(II-KAJPI)4,8,8

```

```

PLSQ1100
PLSQ1110
PLSQ1120
PLSQ1130
PLSQ1140
PLSQ1150
PLSQ1160
PLSQ1170
PLSQ1180
PLSQ1190
PLSQ1200
PLSQ1210
PLSQ1220
PLSQ1230
PLSQ1240
PLSQ1250
PLSQ1260
PLSQ1270
PLSQ1280
PLSQ1290
PLSQ1291
PLSQ1292
PLSQ1293
PLSQ1294
PLSQ1295
PLSQ1296
PLSQ1297
PLSQ1298
PLSQ1300
PLSQ1310
PLSQ1320
PLSQ1330
PFIT 0
PFIT 10
PFIT 20
PFIT 30
PFIT 40
PFIT 50
PFIT 60
PFIT 70
PFIT 80
PFIT 90
PFIT 100
PFIT 110
PFIT 120
PFIT 130
PFIT 140
PFIT 150
PFIT 160
PFIT 170
PFIT 180
PFIT 190
PFIT 200
PFIT 210
PFIT 220

C   COMPUTE NEXT ORDER POLYNOMIAL
4 DD 5 I=1,NCASE
TEMP=POLY(I)
POLY(I)=(X (I)-ALPHA(II))*POLY(I)-8*POLYO(I)
5 POLYO(I)=TEMP
6 8=8ETA(II)
C   OUTPUT PREDICTED VALUES
8 LINES = 50
IF(IPR)7,9,7
7 WRITE(ITW,103)
9 DO 12 I=1,NCASE
IF(LINES-50) 11,10,10
10 NPAGE = NPAGE & 1
LINES = 0
CALL FMAT(IPR,ITW)
IF(IPR) 13,13,11
13 WRITE(ITW,101) TITLE,IPROB,NPAGE
WRITE(ITW,103)
11 DIF = Y(I) - YA(I)
LINES = LINES & 1
12 WRITE(ITW,102) I,ID(I),X(I),Y(I),YA(I),DIF
RETURN
ENO

// DUP
*STORE      WS UA PFIT

```

```

PFIT 230
PFIT 240
PFIT 250
PFIT 260
PFIT 270
PFIT 280
PFIT 290
PFIT 300
PFIT 310
PFIT 320
PFIT 330
PFIT 340
PFIT 350
PFIT 360
PFIT 370
PFIT 380
PFIT 390
PFIT 400
PFIT 410
PFIT 420
PFIT 430
PFIT 440
PFIT 450
PFIT 460
PFIT 470

```

```

// FOR SUBROUTINE TO COMPUTE POLYNOMIAL DERIVATIVES
* ONE WORD INTEGERS
C SUBROUTINE TO COMPUTE POLYNOMIAL DERIVATIVES
SUBROUTINE POER
COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,ISCR,
INCASE,ICOF,IDER,NDER,IALP,INMD2,KX(5),EPS,FLVB(4),XB,X14
COMMON TITLE(18),ID(150),X(150),Y(150),C(51),ALPHA(51),BETA(51)
COMMON DPOLY(51),DERIV(51),DOPOL(51)
101 FORMAT( 10X18A4,5X3HJOBI7, 5X,4HPAGE,I6)
102 FORMAT(/ 5X,I9,7X,2F15.5,I12,7X,F15.5)
103 FORMAT(61X12,7XF15.5)
104 FORMAT(/5X,14HIDENTIFICATION,14X2HX'14X2HY*8X12HDERIV. ORDER7X12+
IDERIV. VALUE/)
IE(NDER)17,16,17
17 LINES = 50
IF(NF-NDER)30,31,31
30 NDP1=NE+1
GO TO 32
31 NDP1 = NDER + 1
32 KBJP1 = NF + 1
DO 25 IL1=1,NCASE
IE(ID(IL1)) 1,25,25
1 XB = X(IL1)
DO 2 II = 1,KBJP1
DPOLY(II) = 0.0
2 DOPOL(II) = 0.0
DPOLY(KBJP1 + 1) = 0.0
DPOL = 1.0
NN = 1
GO TO 4
C BEGIN COMPUTATION.
3 DPOL = DPOLY(NN)
4 DPOLO = 0.0
DERIV(NN)=0.0
B = 0.0
II = 1
C COMPUTATION OF FITTED VALUE AND DERIVATIVE.
5 DERIV(NN) = DERIV(NN) + C(II) * DPOL
IF(II-KBJP1) 6,7,7
C COMPUTATION OF A NEW POLYNOMIAL DERIVATIVE.
6 TEMP = DPOL
DPOL = (XB - ALPHA(II))*DPOL + (NN - 1) * DOPOL(II) - B*DPOLO
DPOLO = TEMP
DOPOL(II)=DPOLO
B = BETA(II)
II = II + 1
GO TO 5
C COMPUTATION OF THE NEXT DERIVATIVE.
7 IF(NN - NDP1) 8,9,8
8 NN = NN + 1
GO TO 3
C OUTPUT DERIVATIVE
9 IF(LINES-50) 12,11,11
11 NPAGE=NPAGE + 1
CALL FMAT(IPR,ITW)

```

PDER 0  
PDER 10  
PDER 20  
PDER 30  
PDER 40  
PDER 50  
PDER 60  
PDER 70  
PDER 80  
PDER 90  
PDER 100  
PDER 110  
PDER 120  
PDER 130  
PDER 140  
PDER 150  
PDER 160  
PDER 170  
PDER 180  
PDER 190  
PDER 200  
PDER 210  
PDER 220  
PDER 230  
PDER 240  
PDER 250  
PDER 260  
PDER 270  
PDER 280  
PDER 290  
PDER 300  
PDER 310  
PDER 320  
PDER 330  
PDER 340  
PDER 350  
PDER 360  
PDER 370  
PDER 380  
PDER 390  
PDER 400  
PDER 410  
PDER 420  
PDER 430  
PDER 440  
PDER 450  
PDER 460  
PDER 470  
PDER 480  
PDER 490  
PDER 500  
PDER 510  
PDER 520  
PDER 530  
PDER 540

```

IF(IPR) 111,111,112
111 WRITE(ITW,101) TITLE,IPROB,NPAGE
112 WRITE(ITW,104)
LINES = C
12 L=1
LINES = LINES + 2
WRITE(ITW,102) ID(IL1),XB,DERIV(1),L,DERIV(2)
IF(NDP1-3) 25,13,13
13 DO 14 J=3,NDP1
L = J-1
LINES = LINES + 1
14 WRITE(ITW,103) L,DERIV(J)
25 CONTINUE
16 RETURN
END
// DUP
*STORE WS UA PDER

```

PDER 550  
PDER 560  
PDER 570  
PDER 580  
PDER 590  
PDER 600  
PDER 610  
PDER 620  
PDER 630  
PDER 640  
PDER 650  
PDER 660  
PDER 670  
PDER 680  
PDER 690  
PDER 700  
PDER 710

```

// FOR INPUT DATA SUBROUTINE
*ONE WORD INTEGERS
*10CS(CARD,1132PRINTER,0ISK)
*NAME REGR
C INPUT DATA SUBROUTINE
COMMON ICR,ICP,IPR,ITW,IT1,IT2,1PRO8,N,NF,CASES,NPAGE,INMO,1PREO,
11STEP,1CNST,1REAR,KX(11),MX(20),NCO(3), ISEQ,NCASE,NX(10),
2 EFOUT,EFIN,TOL,FLV8(2),KNN
COMMON TITLE(18),VNAME(30),SUMY(30),SO(30),X(30),R(30,30)
COMMON HIGH(30),HLOW(30),MF(50,3)
DEFINE FILE 606(500,65,U,IT1)
101 FORMAT(6I2)
102 FORMAT(I4,4X,18A4)
103 FORMAT(I5I2, 2F4.3,F6.5)
104 FORMAT(20A4)
105 FORMAT( 10X,18A4,5X,3HJ08,17,5X,4HPAGE,I6//11X,19HNUMBER OF VARREGR
11ABLES,16X,12/11X,10HINPUT TYPE , 25X,12/11X,14HSEQUENCE CHECK 21XREGR
212/11X,19HVARIALES ON CARD 1 16X,12/11X19HVARIALES ON CARD 2 16XREGR
312/11X,19HVARIALES ON CARD 3 16X,12/11X,21HTRANSFORMATION SWITCH,REGR
414X,12/11X,25HOUTPUT RAW CROSS PRODUCTS 10X,12/11X,30HOUTPUT RESIDREGR
5UAL CROSS PRODUCTS 5X,12)
106 FORMAT(11X,22HPRINT PREOICTED VALUES,13X,12/11X11HPRINT STEPS,24X,REGR
112/11X,14HPOOLING OPTION,21X,12 /11X,18HDEPENDENT VARIABLE 17REGR
2X,12/11X,27HF-LEVEL TO REMOVE VARIABLES,F10.3/11X,26HF-LEVEL TO ENREGR
3TER VARIABLES,F11.3/11X,15HTOLERANCE VALUE,IIX,F11.5/11X,28HOUTPUTREGR
4 VARIANCE - COVARIANCE 7X,12/11X,18HOUTPUT CORRELATION 17X,I2) REGR
107 EORMAT(///' AN ILLEGAL CHARACTER HAS BEEN ENCOUNTERED IN COLUMN', REGR
113,' DE THE ABOVE EORMAT CARD.'/' CHANGE CARD AND RERUN JOB.') REGR
108 FORMAT(///' AN ILLEGAL CHARACTER HAS BEEN ENCOUNTERED AT APPROXIMAREGR
ITELY COLUMN',13,' OF THE ABOVE DATA CARD.'/' CHANGE OR REMOVE CARDREGR
2 AND PRESS START TO CONTINUE.') REGR
109 FORMAT(//5X4HCARDI10,4H NO.14,1X 30HIS OUT OF SEQUENCE. RERUN JOB,REGR
1 ) REGR
110 FORMAT(///' INVALID INPUT OPTION-JOB TERMINATED ') REGR
KNN=0 REGR
NPAGE = 0 REGR
READ(2,101) ICR,ICP,IPR,ITW,IT1,IT2 REGR
IF(1PR)701,702,701 REGR
702 ITW=3 REGR
GO TO 703 REGR
701 ITW=1 REGR
703 READ(ICR,102) 1PRO8,TITLE REGR
READ(ICR,103) N,INMO,1SEQ,(NCO(1),I=1,3),MX(20), REGR
1(MX(I),I=1,4),1PREO,1STEP,1CNST,1REAR,EFOUT,EFIN,TOL REGR
CALL FMAT(IPR,ITW) REGR
WRITE(ITW,105) TITLE,1PRO8,NPAGE,N,INMO,1SEQ,(NCO(1),I=1,3),MX(20) REGR
1,(MX(1),I=1,2) REGR
WRITE(ITW,106) 1PREO,1STEP,1CNST,1REAR,EFOUT,EFIN,TOL, REGR
1(MX(I),I=3,4) REGR
RFAO(ICR,104) (VNAME(1),I=1,N) REGR
IF(INMO-1) 1002,1,1004 REGR
1004 IF(INMO-4) 5,1002,1002 REGR
1 DO 4 I=1,3 REGR
CALL FMTRD(MF(1,1),IRR) REGR
CALL PRNTB REGR

```

```

REGR 0 IF(IRR) 2,3,2 REGR 550
REGR 10 2 WRITE(ITW,107) IRR REGR 560
REGR 20 CALL EXIT REGR 570
REGR 30 1002 WRITE(ITW,110) REGR 580
REGR 40 CALL EXIT REGR 590
REGR 50 3 IF(NCO(1+1)) 5,5,4 REGR 600
REGR 60 4 CONTINUE REGR 610
REGR 70 C SUBROUTINE TO READ SOURCE DATA REGR 620
REGR 80 C INITIALIZATION REGR 630
REGR 90 5 DO 8 I=1,N REGR 640
REGR 100 HIGH(I) = 0. REGR 650
REGR 110 SO(I)=0. REGR 660
REGR 120 HLOW(I) = 1.0E+36 REGR 670
REGR 130 SUMY(I) = 0 REGR 680
REGR 140 DO 8 J=1,N REGR 690
REGR 150 8 R(I,J) = 0.0 REGR 700
REGR 160 KOUNT = 0 REGR 710
REGR 170 CASES = 0. REGR 720
REGR 180 NCASE = 0 REGR 730
REGR 190 9 IT1 = 1 REGR 740
REGR 200 10 GO TO (11,41,51),INMD REGR 750
REGR 210 C CARD READER INPUT REGR 760
REGR 220 11 IST = 1 REGR 770
REGR 230 I=1 REGR 780
REGR 240 IF(NCO(1)) 12,12,13 REGR 790
REGR 250 12 NCD(I) = N REGR 800
REGR 260 13 CALL DATRD(MF(1,1),IRR,IO,1,NC,1, X,-NCD(1),0,0) REGR 810
REGR 270 IF(IRR) 14,15,14 REGR 820
REGR 280 14 CALL PRNTB REGR 830
REGR 290 WRITE(ITW,108) IRR REGR 840
REGR 300 PAUSE 10 REGR 850
REGR 310 GO TO (13,18,18),I REGR 860
REGR 320 15 IF(10) 100,16,16 REGR 870
REGR 330 16 DO 22 I=2,3 REGR 880
REGR 340 IF(NCO(1)) 23,23,17 REGR 890
REGR 350 17 IST = NCD(1-1) + IST REGR 900
REGR 360 18 CALL DATRD(MF(1,1),IRR,1D1,1,NC1,1,X(IST),-NCD(I),0,0) REGR 910
REGR 370 IF(IRR) 14,19,14 REGR 920
REGR 380 19 IF(1SEQ) 22,22,20 REGR 930
REGR 390 20 IF(1D-1D1) 60,21,60 REGR 940
REGR 400 21 IF(NC1-NC) 60,60,6 REGR 950
REGR 410 6 ID = 101 REGR 960
REGR 420 NC = NC1 REGR 970
REGR 430 22 CONTINUE REGR 980
REGR 440 GO TO 23 REGR 990
REGR 450 60 WRITE(ITW,109) 1D1,NC1 REGR1000
REGR 460 CALL EXIT REGR1010
REGR 470 23 IF(MX(20)) 230,231,230 REGR1020
REGR 480 230 CALL TRAN REGR1030
REGR 490 231 IF(INMO-1) 1002,27,30 REGR1040
REGR 500 27 WRITE(606,IT1) 1D , (X(1),I=1,N) REGR1050
REGR 510 C COMPUTE CROSS PRODUCT MATRIX REGR1060
REGR 520 30 CASES = CASES + 1. REGR1070
REGR 530 NCASE = NCASE + 1 REGR1080
REGR 540 DO 35 I = 1,N REGR1090

```

```

SUMY(I) = SUMY(I) + X(I)
DO 35 J=I,N
  R(I,J) = R(I,J) + X(I)*X(J)
35 R(J,I) = R(I,J)
C   DETERMINE HIGH AND LOW VALUFS
DO 39 I=1,N
  IF(X(I) - HIGH(I)) 37,37,36
36 HIGH(I) = X(I)
37 IF(X(I) - HLOW(I))38,39,39
38 HLOW(I) = X(I)
39 CONTINUE
GO TO 10
41 READ(606*IT1) IO , (X(I),I=1,N)
  IF(IO) 100,100,23
C   READ A MATRIX FROM CARDS
51 CALL MXRAD
100 IF(INMD-1) 1002,150,151
150 WRITE(606*IT1) IO , (X(I),I=1,N)
151 IT1 = 1
200 IF(NCASE)571,571,572
572 CALL LINK(COREL)
571 CALL LINK(RFGR2)
FNO
// DUP
*STORF      WS UA REGR

```

```

REGR1100 // FOR CALLING PROGRAM FOR CORRELATION AND REGRFSSION
REGR1110 *IOCS(CARD,1132PRINTER,DISK)
REGR1120 *ONE WORD INTEGERS
REGR1130 *NAME REGR2
REGR1140 COMMON ICR,ICP,IPR,ITW,IT1,IT2,IRROB,N,NF,CASES,NPAGE,INMO,IPRFD,
REGR1150 11STEP,ICNST,IREAR,KX(1),MX(20),NCO1,NCO2,NCO3,ISFQ,NCASE,NX(10),
RFR1160 2 EFOUT,FFIN,TOL,FLVB(2),KNN
REGR1170 COMMON TITLE(18),VNAME(30),SUMY(30),SO(30),X(30),R(30,30)
REGR1180 COMMON HIGH(30),HLOW(30)
REGR1190 DEFINE FILE 606(500,65,U,IT1)
REGR1200 100 FORMAT(//2X13HJOB COMPLETFO)
RFR1210 IF(ISTER)4,4,2
RFR1220 2 IF(IREAR) 4,4,3
RFR1230 3 CALL RFGRE
RFR1240 4 WRITE(1TW,100)
RFR1250 CALL EXIT
RFR1260 ENO
REGR1270
REGR1280
REGR1290
REGR1300
REGR1310
RFR1320
REGR1330
REGR1340
// DUP
*STORE      WS UA REGR2
RGR2 0
RGR2 10
RGR2 20
RGR2 30
RGR2 40
RGR2 50
RGR2 60
RGR2 70
RGR2 80
RGR2 90
RGR2 100
RGR2 110
RGR2 120
RGR2 130
RGR2 140
RGR2 150
RGR2 160
RGR2 170
RGR2 180

```

```

// FOR SUBROUTINE FOR STEPWISE REGRESSION
*ONE WORD INTEGERS
C SUBROUTINE FOR STEPWISE REGRESSION
SUBROUTINE REGRE
DIMENSION INDEX(30)
COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPR08,N,NF,CASES,NPAGE,INMD,IPRED,
1 ISTEP,ICNST,IREAR,KX(1),MX(20),NX(15),EFOUT,EFIN,TOL,FLV8(2),KNM
COMMON TITLE(18),VNAME(30),SUMY(30),SD(30),DATA(30),R(30,30)
COMMON CORRY(30),COENI(30)
101 FORMAT( 10X18A4,5X3HJ08I7, 5X,4HPAGE 16)
102 FORMAT(//5X,19HREGRESSION ANALYSIS //5X,18HDEPENDENT VAR(A8LE ,16XRGRE 100
1,A4/5X,27HRESIDUAL STANDARD DEVIATIONF11.4/5X,26HSTANDARD ERROR OFRGRE 110
2 THE MEAN F12.4/5X,10HMULTIPLE R 18X,F10.4/5X,13HMULTIPLE RSQR 15XRGRE 120
3,F10.4)
103 FORMAT(//4X,16HVARIA8LE REMOVED 18X,A4//)
104 FORMAT(//4X,16HVARIA8LE ENTERED 18X,A4//)
105 FORMAT(//3X,8HVARIA8LE11X,8H8 - COEF 4X,14HSTD ERROR OF 8 9X,
19HPARTIAL-R,8X,9HBETA-COEF 4X17HSTD ERROR OF 8ETA//)
106 FORMAT(//30X26HANALYSIS OF VAR(ANCE TABLE)
107 FORMAT(5X,A4,6X,2F15.4,13X,F7.4,2X,F15.4,3X,F15.4)
108 FORMAT(// 2X,8HCONSTANT 4X,F15.4)
109 FORMAT(// 33X,16HPREDICTED VALUES//11X,4HCASE,12X,6HACTUAL,11X,
19HPREDICTED,16X,8HRESIDUAL//)
110 FORMAT(10X,I5,3(5X,E15.4))
111 FORMAT(15X6HSOURCE13X4H0.F.05X14HSUM OF SQUARES3X11HMEAN SQUAREO8RGRE 240
1X1HF)
112 FORMAT(/// 42HMEAN SQUARE NON-POSITIVE. JOB TERMINATED. )
113 FORMAT (///41H NO MORE DEGREES FREEDOM. JOB TERMINATED. )
114 FORMAT( 15X10HREGRESS(ONSX,I6,5X,E14.5,E16.5,E15.5)
115 FORMAT( /// 62HNO MORE VARIABLES SATISFY VARIANCE CRITERION. JOB TRRGRE 290
TERMINATED. )
116 FORMAT(15X5HERROR10X,I6,5X,E14.5,E16.5)
117 FORMAT(15X4HMEAN11X,I6,5X,E14.5,E16.5)
PLACE DEPENDENT VARIAB8LE AT END OF MATRIX
IKL=1
IKT=0
IF(IREAR) 6,6,2
2 DO 3 I = 1,N
T = RII,N)
RII,N) = R(I,IREAR)
3 RII,IREAR) = T
DO 4 I=1,N
T = R(N,I)
R(N,I) = R(IREAR,I)
4 R(IREAR,I) = T
T = SUMY(N)
SUMY(N) = SUMY(IREAR)
SUMY(IREAR) = T
T = SD(N)
SD(N) = SD(IREAR)
SD(IREAR) = T
T = VNAME(N)
VNAME(N) = VNAME(IREAR)
VNAME(IREAR) = T
INITIALIZE COMPUTATIONAL PARAMETERS
RGRE 0
RGRE 10
RGRE 20
RGRE 30
RGRE 40
RGRE 50
RGRE 60
RGRE 70
RGRE 80
RGRE 90
RGRE 100
RGRE 110
RGRE 120
RGRE 130
RGRE 140
RGRE 150
RGRE 160
RGRE 170
RGRE 180
RGRE 190
RGRE 200
RGRE 210
RGRE 220
RGRE 230
RGRE 240
RGRE 250
RGRE 260
RGRE 270
RGRE 280
RGRE 290
RGRE 300
RGRE 310
RGRE 320
RGRE 330
RGRE 340
RGRE 350
RGRE 360
RGRE 370
RGRE 380
RGRE 390
RGRE 400
RGRE 410
RGRE 420
RGRE 430
RGRE 440
RGRE 450
RGRE 460
RGRE 470
RGRE 480
RGRE 490
RGRE 500
RGRE 510
RGRE 520
RGRE 530
RGRE 540

6 NOVMI = N-1
DO 7 I = 1,NOVMI
7 CORRY(I) = R(I,N)
DEFR = CASES - 1.
SSM=CASES*SUMY(N)**2
SSY=DEFR *SD(N)**2
ANODA=SQRT(DEFR/CASES)
NOENT = 0
NOMIN = 0
NOMAX = 0
C START OF MAIN ITERATION FOR A VAR(A8LE
C COMPUTE STANDARD ERROR OF MEAN AND ESTIMATE
11 SMEAN = SD(N)*SQRT(R(N,N)/DEFR)*ANODA
SEST = SMEAN * SQRT(CASES)
C COMPUTE MULTIPLE R AND MULTIPLE R**2
R2M1 = 1.0-R(N,N)
IF(R2M1) 31,31,30
30 RMLT = SQRT(R2M1)
GO TO 32
31 RMLT = 0.0
32 RSQ = RMLT**2
C INITIALIZE VARIAB8LE ENTRY PARAMETERS
VMIN = 1.0E20
VMAX = 0.0
VAR = 0.0
NOIN = 0
C DETERMINE ENTRY VARIAB8LES AND COMPUTE COEFFICIENTS
C AND THEIR STANDARD ERRORS
35 DO 56 I=1,NOVMI
41 IF(R(I,I) - TOL) 56,42,42
42 VAR = R(I,N)*R(N,I)/RII,I)
IF(VAR) 44,56,53
44 NOIN = NOIN & 1
INDEX(NOIN) = I
IF (ABS(VAR) - ABSIVMIN)) 50, 50, 56
50 VMIN = VAR
NOMIN = I
52 GO TO 56
53 IF(VAR-VMAX) 56,56,54
54 VMAX = VAR
NOMAX = I
56 CONTINUE
C IF NO VARIAB8LES ENTERED GO TO NEXT ITERATION
IF(NOIN) 82,82,66
C OUTPUT REGRESSION EQUATION FOR THIS STEP
66 IF(ISTEP)400,401,400
400 IF(ISTEP-NOIN) 68,68,78
68 NPAGE = NPAGE & 1
CALL FMAT(IPR,ITW)
IF(IPR) 681,681,682
681 WRITE(ITW,101) TITLE,IPR08,NPAGE
682 WRITE (ITW,102) VNAMEIN),SEST,SMEAN,RMLT,RSQ
(F(NDENT) 69,69,71
69 WRITE(ITW,103) VNAMEIK)
GO TO 72
RGRE 550
RGRE 560
RGRE 570
RGRE 580
RGRE 590
RGRE 600
RGRE 610
RGRE 620
RGRE 630
RGRE 640
RGRE 650
RGRE 660
RGRE 670
RGRE 680
RGRE 690
RGRE 700
RGRE 710
RGRE 720
RGRE 730
RGRE 740
RGRE 750
RGRE 760
RGRE 770
RGRE 780
RGRE 790
RGRE 800
RGRE 810
RGRE 820
RGRE 830
RGRE 840
RGRE 850
RGRE 860
RGRE 870
RGRE 880
RGRE 890
RGRE 900
RGRE 910
RGRE 920
RGRE 930
RGRE 940
RGRE 950
RGRE 960
RGRE 970
RGRE 980
RGRE 990
RGRE1000
RGRE1010
RGRE1020
RGRE1030
RGRE1040
RGRE1050
RGRE1060
RGRE1070
RGRE1080
RGRE1090

```

```

71 WRITE(ITW,104) VNAME(K)
72 WRITE(ITW,105)
63 CNST = SUMY(N)
65 DD 76 I=1,NOIN
   IL = INDEX(I)
   PARTL = D.0
C   COMPUTE CDEFFICIENTS AND THEIR STANDARD ERRORS
   BETA = R(IL,N)
   COEN(I) = BETA*SD(N)/SD(IL)
   BER = SQRT(R(N,N)*R(IL,IL)/DEFR)
   SIGM = 8ER*SD(N)/SD(IL)
C   COMPUTE PARTIAL CORRELATION COEFFICIENTS
   DD 58 J = 1,NOIN
   JL = INDEX(J)
58 PARTL=PARTL+(R(JL,N)-R(IL,N) *R(JL,IL)/R(IL,IL))*CORRY(JL)
   PARTL = SIGN(SQRT(1.0-R(N,N)/(1.0-PARTL)),COEN(I))
   WRITE(ITW,107) VNAME(IL),COEN(I),SIGM,PARTL,8ETA,8ER
C   COMPUTE CONSTANT TERM
76 CNST = CNST-(COEN(I)*SUMY(IL))
   WRITE(ITW,108) CNST
   WRITE(ITW,106)
   WRITE(ITW,111)
   IDF=CASES-DEFR-1.
   IEDF=DEFR
   AMSE=SEST**2
   SSE=AMSE*DEFR
   SSR=SSY-SSE
   AMSR=SSR/(CASES-DEFR-1.)
   AF=AMSR/AMSE
   WRITE(ITW,117) IK1,SSM,SSM
   WRITE(ITW,114) IDF,SSR,AMSR,AF
   WRITE(ITW,116) IEDF,SSE,AMSE
C   PRINT PREDICTED VALUES AND RESIDUALS
78 IF(INMD-3)178,79,178
178 IF(IPRED)79,79,251
251 IF(IPRED-ISTEP)79,151,151
151 IF(NOIN-IPRED) 79,77,77
77 LIZ = 40
   IT1 = 1
16 READ(606,IT1) IO ,(DATA(I),I=1,N)
   IF (IO) 79,79,17
17 IF(IREAR) 19,19,18
18 T = DATA(N)
   DATA(N) = DATA(IREAR)
   DATA(IREAR) = T
19 YPRED = CNST
   DD 22 I=1,NDIN
   KK = INDEX(I)
22 YPRED = YPRED & COEN(I) * DATA(KK)
   DEV = DATA(N) - YPRED
   IF(LIZ-40) 25,24,24
24 LIZ = 0
   NPAGE = NPAGE & 1
   CALL FMAT(IPR,ITW)
   IF(IPR) 241,241,242

```

```

RGRE1100
RGRE1110
RGRE1120
RGRE1130
RGRE1140
RGRE1150
RGRE1160
RGRE1170
RGRE1180
RGRE1190
RGRE1200
RGRE1210
RGRE1220
RGRE1230
RGRE1240
RGRE1250
RGRE1260
RGRE1270
RGRE1280
RGRE1290
RGRE1300
RGRE1310
RGRE1320
RGRE1330
RGRE1340
RGRE1350
RGRE1360
RGRE1370
RGRE1380
RGRE1390
RGRE1400
RGRE1410
RGRE1420
RGRE1430
RGRE1440
RGRE1450
RGRE1460
RGRE1470
RGRE1480
RGRE1490
RGRE1500
RGRE1510
RGRE1520
RGRE1530
RGRE1540
RGRE1550
RGRE1560
RGRE1570
RGRE1580
RGRE1590
RGRE1600
RGRE1610
RGRE1620
RGRE1630
RGRE1640
241 WRITE(ITW,101) TITLE,IPRO8,NPAGE
242 WRITE(ITW,109)
25 LIZ = LIZ & 1
26 WRITE(ITW,110) ID,DATA(N),YPRED,DEV
   GO TO 16
C   IF VARIANCE CDNTRIBUTION INSIGNIFICANT - REMDVE VARIABLE K
79 IF(IKT)179,179,401
179 FLEV = ABS(VMIN) * DEFR / R(N,N)
   IF (FLEV - EFOUT) 80,82,82
80 K = NDMIN
   DEFR = DEFR & 1.0
   NOENT = 0
   GO TO 89
C   IF VARIANCE CONTRIBUION SIGNIFICANT - ENTER VARIABLE K
82 DENOM = R(N,N) - VMAX
   IF(DENOM) 210,210,84
84 FLEV = VMAX * DEFR / DENOM
   IF (FLEV - EFIN) 402,402,87
87 K = NOMAX
   NOENT = K
   DEFR=DEFR-1.0
C   IF DEGREES DF FREEDOM NON-POSITIVE,TERMINATE JOB
   IF(DEFR) 34,89,89
34 WRITE(ITW,113)
401 RETURN
C   IF VARIANCE CRITERION NOT SATISFIED - TERMINATE JOB
89 IF(K) 90,90,92
90 WRITE(ITW,115)
   GO TO 401
C   REARRANGE INVERSE FOR ENTERING OR DELETING A VARIABLE
92 DD 98 I=1,N
   IF(I-K) 94,98,94
94 DD 97 J=1,N
   IF(J-K) 96,97,96
96 R(I,J) = R(I,J) - {R(I,K)*R(K,J)/R(K,K)}
97 CDNTINUE
98 CONTINUE
   DD 202 I=1,N
   IF(I-K) 201,202,201
201 R(I,K) = -R(I,K)/R(K,K)
202 CDNTINUE
   DD 206 J=1,N
   IF(J-K) 205,206,205
205 R(K,J) = R(K,J)/R(K,K)
206 CDNTINUE
   R(K,K) = 1.0/R(K,K)
C   TEST FOR POSITIVE MEAN SQUARE
   IF(R(N,N)) 210,210,11
210 WRITE(ITW,112)
   GO TO 401
402 IKT=1
   IF(INMD-3)404,401,404
404 IF(IPRED)77,401,401
   END
// DUP

```

```

RGRE1650
RGRE1660
RGRE1670
RGRE1680
RGRE1690
RGRE1700
RGRE1710
RGRE1720
RGRE1730
RGRE1740
RGRE1750
RGRE1760
RGRE1770
RGRE1780
RGRE1790
RGRE1800
RGRE1810
RGRE1820
RGRE1830
RGRE1840
RGRE1850
RGRE1860
RGRE1870
RGRE1880
RGRE1890
RGRE1900
RGRE1910
RGRE1920
RGRE1930
RGRE1940
RGRE1950
RGRE1960
RGRE1970
RGRE1980
RGRE1990
RGRE2000
RGRE2010
RGRE2020
RGRE2030
RGRE2040
RGRE2050
RGRE2060
RGRE2070
RGRE2080
RGRE2090
RGRE2100
RGRE2110
RGRE2120
RGRE2130
RGRE2140
RGRE2150
RGRE2160
RGRE2170
RGRE2180
RGRE2190

```

\*STORE WS UA REGRE

RGRE2200

```

// FOR SUBROUTINE TO READ PARAMETER CARDS AND DATA NOVA 0
*ONE WORD INTEGERS NOVA 10
*LOCS(CARD,1132PRINTER,DISK) NOVA 20
*NAME ANOVA NOVA 30
C SUBROUTINE TO READ PARAMETER CARDS AND DATA NOVA 40
COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,NPAGE,INMD,NF,ITRN,NA,NB,NC, NOVA 50
IND,TITLE(18),NX(5),LS(5),IN(4),NDIV(20),SMQR(20),XDEV(20),X(1500) NOVA 60
DEFINE FILE 606(500,6,U,IT1) NOVA 70
DEFINE FILE 607(1000,2,U,IT2) NOVA 80
101 FORMAT (7I2) NOVA 90
102 FORMAT ( 10X,18A4,5X,3HJOB,I7,5X,4HPAGE,I6///10X,17HNUMBER OF FNOVA 100
FACTORS,15X,I2/ 10X,10HINPUT MODE,22X,I2/ 10X,21HTRANSFORMATION SWINOVA 110
2TCH,11X,I2/ 10X,27HNUMBER OF LEVELS - FACTOR 1,5X,I2/ 10X,27HNUMBERNOVA 120
3R OF LEVELS - FACTOR 2,5X,I2/ 10X,27HNUMBER OF LEVELS - FACTOR 3, NOVA 130
45X,I2/ 10X,27HNUMBER OF LEVELS - FACTOR 4,5X,I2) NOVA 140
103 FORMAT(6I2) NOVA 150
104 FORMAT(I4,4X,18A4) NOVA 160
107 FORMAT(///' AN ILLEGAL CHARACTER HAS BEEN ENCOUNTERED IN COLUMN', NOVA 170
I13,' OF THE ABOVE FORMAT CARD.///' CHANGE CARD AND RERUN JOB.')
```

NOVA 180

```

108 FORMAT(///' AN ILLEGAL CHARACTER HAS BEEN ENCOUNTERED AT APPROXIHANOVA 190
ITELY COLUMN',I3,' OF THE ABOVE DATA CARD.///' CHANGE OR REMOVE CARDNOVA 200
2 AND PRESS START TO CONTINUE.')
```

NOVA 210

```

109 FORMAT(///' INVALID INPUT OPTION-JOB TERMINATED ')
```

NOVA 220

```

C READ PARAMETER CARDS NOVA 230
NPAGE=0 NOVA 240
READ(2,103) ICR,ICP,IPR,ITW,IT1,IT2 NOVA 250
IF(IPR)701,702,701 NOVA 260
702 ITW=3 NOVA 270
GO TO 703 NOVA 280
701 ITW=1 NOVA 290
703 READ(ICR,104) IPROB,TITLE NOVA 300
READ (ICR,101) NF,INMD,ITRN,(NX(I),I=1,4) NOVA 310
CALL FMAT(IPR,ITW) NOVA 320
WRITE (ITW,102) TITLE,IPROB,NPAGE,NF,INMD,ITRN,(NX(I),I=1,4) NOVA 330
IF(INMD-1) 1,1,3 NOVA 340
1 CALL FMTRD(NOIV,IRR) NOVA 350
CALL PRNTB NOVA 360
IF(IRR) 2,5,2 NOVA 370
2 WRITE(ITW,107) IRR NOVA 380
CALL EXIT NOVA 390
C SUBROUTINE TO READ SOURCE DATA (ANALYSIS OF VARIANCE) NOVA 400
3 IF((NMD-2)5,5,4) NOVA 410
4 WRITE(ITW,109) NOVA 420
CALL EXIT NOVA 430
5 NA = NX(1) + 1 NOVA 440
NB = NX(2) + 1 NOVA 450
NC = NX(3) + 1 NOVA 460
ND = NX(4) + 1 NOVA 470
LS(1)=1 NOVA 480
LS(2) = NA NOVA 490
LS(3) = LS(2) * NB NOVA 500
LS(4) = LS(3) * NC NOVA 510
LS(5)=LS(4)*ND NOVA 520
J=1 NOVA 530
189 GO TO (10,40),INMD NOVA 540
```

```

C      READ DATA FROM CARD READER
10 CALL DATRO(NDIV,IRR,IN(1),4,DATA,-1,0,0)
   WRITE(606*J)  (IN(I),I=1,4),DATA
   J=J+1
   IF(IRR) 17,18,17
17 CALL PRNTB
   WRITE(ITW,108) IRR
   PAUSE 10
   GO TO 10
18 IF(ITRN) 19,20,19
19 CALL TRAN
20 IF (IN(1)) 50,50,21
21 IS=IN(1)
   DO 30 I=2,NF
30 IS=IS+LS(I)*(IN(I)-1)
32 CALL STORE (DATA,IS)
   GO TO 189
C      READ DATA FROM DISC OR TAPE
40 READ(606*J)  (IN(I),I=1,4),DATA
   J=J+1
   GO TO 18
50 CALL LINK(ANOV2)
   END
// DUP
*STORE      WS  UA  ANOVA

```

```

NOVA 550 // FOR SUBROUTINE TO STORE A DATUMIN CORE OR DISC
NOVA 560 *ONE WORD INTEGERS
NOVA 570 C SUBROUTINE TO STORE A DATUMIN CORE OR DISC
NOVA 580 SUBROUTINE STORE (DATA,IS)
NOVA 590 COMMON ICR,IRP,IPR,ITW,IT1,IT2,IPR08,NPAGE,INMD,NF,ITRN,NA,N8,NC,
NOVA 600 IND,TITLE(18),NX(5),LS(5),IN(4),NDIV(20),SMQR(20),XDEV(20),X(1500)
NOVA 610 IF (IS-1500) 10,10,20
NOVA 620 10 X(IS)=DATA
NOVA 630 GO TO 30
NOVA 640 C WRITE DATA ON DISC AT LOCATION IS-1500
NOVA 650 20 IST=IS-1500
NOVA 660 WRITE(607*IST) DATA
NOVA 670 30 RETURN
NOVA 680 END
NOVA 690 // DUP
NOVA 700 *STORE      WS  UA  STORE
NOVA 710
NOVA 720
NOVA 730
NOVA 740
NOVA 750
NOVA 760
NOVA 770
NOVA 780
NOVA 790

```

```

STOR 0
STOR 10
STOR 20
STOR 30
STOR 40
STOR 50
STOR 60
STOR 70
STOR 80
STOR 90
STOR 100
STOR 110
STOR 120
STOR 130
STOR 140
STOR 150

```

```

// FOR SUBROUTINE TO GET A DATUM FROM CORE OR DISC
*ONE WORD INTEGERS
C   SUBROUTINE TO GET A DATUM FROM CORE OR DISC
      SUBROUTINE GET (DATA,IS)
      COMMON ICR,IRP,IPR,ITW,IT1,IT2,IPROB,NPAGE,INMD,NF,ITRN,NA,NB,NC,
      INO,TITLE(18),NX(5),LS(5),IN(4),NDIV(20),SMQR(20),XOEV(20),X(1500)
      IF (IS-1500) 10,10,20
10  DATA=X(IS)
      GO TO 30
C   GET DATA FROM DISC AT LOCATION IS-1500
20  IST=IS-1500
      READ(607,IT2) DATA
30  RETURN
      END
// OUP
*STORE      WS  UA  GET

```

```

GETO  0
GETO 10
GETO 20
GETO 30
GETO 40
GETO 50
GETO 60
GETO 70
GETO 80
GETO 90
GETO 100
GETO 110
GETO 120
GETO 130
GETO 140
GETO 150

```

```

// FOR SECONDARY MAIN PROGRAM - ANALYSIS OF VARIANCE
*ONE WORD INTEGERS
*IOCS(CARD,1132PRINTER,OISK)
*NAME ANOV2
C   SECONDARY MAIN PROGRAM - ANALYSIS OF VARIANCE
      COMMON ICR,IRP,IPR,ITW,IT1,IT2,IPROB,NPAGE,INMD,NF,ITRN,NA,NB,NC,
      INO,TITLE(18),NX(5),LS(5),IN(4),NOIV(20),SMQR(20),XOEV(20),X(2000)
      OEFINE FILE 606(500,6,U,IT1)
      OEFINE FILE 607(1000,2,U,IT2)
100 FORMAT(/2X13HJOB COMPLETED)
      CALL SDOP
      CALL MNSQ
      CALL REPT
      WRITE(ITW,100)
      CALL EXIT
      END
// OUP
*STORE      WS  UA  ANOV2

```

```

NOV2  0
NOV2 10
NOV2 20
NOV2 30
NOV2 40
NOV2 50
NOV2 60
NOV2 70
NOV2 80
NOV2 90
NOV2 100
NOV2 110
NOV2 120
NOV2 130
NOV2 140
NOV2 150
NOV2 160
NOV2 170

```

```

// FOR SUBROUTINE TO PERFORM SIGMA AND DELTA OPERATIONS
*ONE WORD INTEGERS
C SUBROUTINE TO PERFORM SIGMA AND DELTA OPERATIONS
  SUBROUTINE SOOP
    COMMON ICR,IRP,IPR,ITW,IT1,IT2,IPROB,NPAGE,INMO,NF,ITRN,NA,NB,NC,
    INO,TITLE(18),NX(5),LS(5),IN(4),NOIV(20),SMQR(20),XDEV(20),X(1500)
60 NFP1=NF+1
    DO 130 K=1,NF
      NN=NX(K)
      FN=NN
      IS=1
      ISPM=1
70 SUMX=0.
      DO 80 I=1,NN
        CALL GET (DATA,IS)
        SUMX=SUMX+DATA
80 IS=IS+LS(K)
        CALL STORE (SUMX,IS)
        DO 90 I=1,NN
          CALL GET (DATA,ISPM)
          DATA=FN*DATA-SUMX
          CALL STORE (DATA,ISPM)
90 ISPM=ISPM+LS(K)
          ITEST= IS-LS(NF+1)
          IF (ITEST) 100,130,130
100 IF (ITEST+LS(K)) 110,110,120
110 IS=IS+LS(K)
          ISPM=ISPM+LS(K)
          GO TO 70
120 IS=ITEST+LS(K)+1
          ISPM=ISPM+LS(K)+1-LS(NF+1)
          GO TO 70
130 CONTINUE
      RETURN
      END
// DUP
*STORE WS UA SOOP

```

```

SOOP 0
SOOP 10
SOOP 20
SOOP 30
SOOP 40
SOOP 50
SOOP 60
SOOP 70
SOOP 80
SOOP 90
SOOP 100
SOOP 110
SOOP 120
SOOP 130
SOOP 140
SOOP 150
SOOP 160
SOOP 170
SOOP 180
SOOP 190
SOOP 200
SOOP 210
SOOP 220
SOOP 230
SOOP 240
SOOP 250
SOOP 260
SOOP 270
SOOP 280
SOOP 290
SOOP 300
SOOP 310
SOOP 320
SOOP 330
SOOP 340
SOOP 350
SOOP 360

```

```

// FOR SUBROUTINE TO COMPUTE MEAN SQUARE SUMMARYS
*ONE WORD INTEGERS
C SUBROUTINE TO COMPUTE MEAN SQUARE SUMMARYS
  SUBROUTINE MNSQ
    COMMON ICR,IRP,IPR,ITW,IT1,IT2,IPROB,NPAGE,INMO,NF,ITRN,NA,NB,NC,
    INO,TITLE(18),NX(5),LS(5),IN(4),NOIV(20),SMQR(20),XDEV(20),X(1500)
C CLEAR SUMMARY TABLE
    DO 140 I=1,15
      NOIV(I)=0
140 SMQR(I)=0.0
      IA=1
      IB=1
      IC=1
      IO=1
      I=0
      GO TO 160
150 CALL GET (DATA,I)
      SMQR(K)=SMQR(K)+DATA**2
      XDEV(K)=DATA
      NOIV(K)=NOIV(K)+1
160 I=I+1
      IF (IA-NA) 170,320,320
170 IA=IA+1
      IF (IB-NB) 180,250,250
180 IF (IC-NC) 190,220,220
190 IF (IO-NO) 200,210,210
200 K=15
      GO TO 150
210 K=11
      GO TO 150
220 IF (IO-NO) 230,240,240
230 K=12
      GO TO 150
240 K=5
      GO TO 150
250 IF (IC-NC) 260,290,290
260 IF (IO-NO) 270,280,280
270 K=13
      GO TO 150
280 K=6
      GO TO 150
290 IF (IO-NO) 300,310,310
300 K=7
      GO TO 150
310 K=1
      GO TO 150
320 IA=1
      IF (IB-NB) 330,400,400
330 IB=IB+1
      IF (IC-NC) 340,370,370
340 IF (IO-NO) 350,360,360
350 K=14
      GO TO 150
360 K=8
      GO TO 150

```

```

MNSQ 0
MNSQ 10
MNSQ 20
MNSQ 30
MNSQ 40
MNSQ 50
MNSQ 60
MNSQ 70
MNSQ 80
MNSQ 90
MNSQ 100
MNSQ 110
MNSQ 120
MNSQ 130
MNSQ 140
MNSQ 150
MNSQ 160
MNSQ 170
MNSQ 180
MNSQ 190
MNSQ 200
MNSQ 210
MNSQ 220
MNSQ 230
MNSQ 240
MNSQ 250
MNSQ 260
MNSQ 270
MNSQ 280
MNSQ 290
MNSQ 300
MNSQ 310
MNSQ 320
MNSQ 330
MNSQ 340
MNSQ 350
MNSQ 360
MNSQ 370
MNSQ 380
MNSQ 390
MNSQ 400
MNSQ 410
MNSQ 420
MNSQ 430
MNSQ 440
MNSQ 450
MNSQ 460
MNSQ 470
MNSQ 480
MNSQ 490
MNSQ 500
MNSQ 510
MNSQ 520
MNSQ 530
MNSQ 540

```

```

370 IF (ID-ND) 380,390,390
380 K=9
   GD TD 150
390 K=2
   GD TD 150
400 IB=1
   IF (IC-NC) 410,440,440
410 IC=IC+1
   IF (ID-ND) 420,430,430
420 K=10
   GD TD 150
430 K=3
   GD TD 150
440 IC=1
   IF (ID-ND) 450,460,460
450 ID=ID+1
   K=4
   GD TD 150
460 CALL GET (DATA,I)
   SMQR(16)=DATA**2
   XDEV(16)=DATA**
   RETURN
   END
// DUP
*STDRS      WS UA MNSQ

```

```

MNSQ 550 // FOR SUBROUTINE TO GENERATE ANALYSIS OF VARIANCE TABLES RPRT 0
MNSQ 560 *DNE WORD INTEGERS RPRT 10
MNSQ 570 C SUBROUTINE TO GENERATE ANALYSIS OF VARIANCE TABLES RPRT 20
MNSQ 580 SUBROUTINE REPRT RPRT 30
MNSQ 590 COMMON ICR,IRP,IPR,ITW,IT1,IT2,IPROB,NPAGE,INMD,NF,ITRN,NA,NB,NC, RPRT 40
MNSQ 600 IND,TITLE(18),NX(5),LS(5),IN(4),NDIV(20),SMQR(20),XDEV(20),X(1500) RPRT 50
MNSQ 610 COMMON NDF(15),HFAD(4),INX(15) RPRT 60
MNSQ 620 101 FORMAT I 9X,18A4,5X,3HJDB,17,5X,4HPAGE,16) RPRT 70
MNSQ 630 102 FORMAT(/// 10X, RPRT 80
MNSQ 640 13CHANALYSIS OF VARIANCE TABLE FOR,1X,I3,3H X ,I3,3H X ,I3, RPRT 90
MNSQ 650 23H X ,I3,11H EXPERIMENT,/// 39X,6H SUM OF,5X,10H DEGREES OF,14X, RPRT 100
MNSQ 660 34H MEAN/13X,9H COMPONENT,17X,7H SQUARES,5X,7H FREEDOM,15X,6H SQUARE///) RPRT 110
MNSQ 670 103 FORMAT (4A4,I4,15I2) RPRT 120
MNSQ 680 104 FORMAT I10X,4A4,4X,F15.2,6X,I5,7X,F15.2) RPRT 130
MNSQ 690 105 FORMAT(/18X,8H RESIDUAL,4X,F15.2,6X,I5,7X,F15.2) RPRT 140
MNSQ 700 106 FORMAT(/21X, 5HTOTAL,4X,F15.2,6X,I5) RPRT 150
MNSQ 710 C FORM DEGREES OF FREEDOM VECTOR FOR COMPONENT MEAN SQUARE RPRT 160
MNSQ 720 NDF(1)=NX(1)-1 RPRT 170
MNSQ 730 NDF(2)=NX(2)-1 RPRT 180
MNSQ 740 NDF(3)=NX(3)-1 RPRT 190
MNSQ 750 NDF(4)=NX(4)-1 RPRT 200
MNSQ 760 NDF(5)= NDF(1)*NDF(2) RPRT 210
MNSQ 770 NDF(6)= NDF(1)*NDF(3) RPRT 220
MNSQ 780 NDF(7)= NDF(1)*NDF(4) RPRT 230
MNSQ 790 NDF(8)= NDF(2)* NDF(3) RPRT 240
NDF(9)= NDF(2)*NDF(4) RPRT 250
NDF(10)= NDF(3)*NDF(4) RPRT 260
NDF(11)= NDF(5)*NDF(3) RPRT 270
NDF(12)= NDF(5)*NDF(4) RPRT 280
NDF(13)= NDF(6)*NDF(4) RPRT 290
NDF(14)= NDF(8)*NDF(4) RPRT 300
NDF(15)= NDF(11)*NDF(4) RPRT 310
C COMPUTE DIVISOR AND INITIALIZE COUNTERS RPRT 320
NN = 1 RPRT 330
DD 6 I = 1,NF RPRT 340
6 NN = NN *NX(I) RPRT 350
FN = NN RPRT 360
KTDFR = NN - 1 RPRT 370
TDTL = 0.0 RPRT 380
NDFRT = 0 RPRT 390
C COMPUTE TDTL SUM OF SQUARES FOR ALL COMPONENTS RPRT 400
TOT = 0.0 RPRT 410
DD 9 I = 1,15 RPRT 420
IF INDIV(I) 9,9,85 RPRT 430
85 SMQR(I) = SMQR(I) / (NDIV(I)*FN) RPRT 440
TOT = TOT + SMQR(I) RPRT 450
9 CONTINUE RPRT 460
C READ LINE CARD AND PRINT COMPONENT RPRT 470
KSW=0 RPRT 480
8 READ (ICR,103) (HEAD(I),I=1,4),INDI,(INX(I),I=1,15) RPRT 490
SMSQ = 0.0 RPRT 500
NDF1 = 0 RPRT 510
C COMPUTE COMPONENT SUM OF SQUARES AND MEAN SQUARE RPRT 520
DD 20 I = 1,15 RPRT 530
IF (INX(I)) 30,30,10 RPRT 540

```

```

10 K=INX(I)
   SMSQ = SMSQ + SMQR(K)
20 NDF1=NDF1+NOF(K)
30 SMSQM=SMSQ/NDF1
C   WRITE TITLE LINE AND COLUMN HEADINGS
   IF(INDI) 40,40,31
31 NPAGE=NPAGE+1
   CALL FMAT(IPR,ITW)
   IF(IPR) 32,32,40
32 WRITE (ITW,101) TITLE,IPR08,NPAGE
40 IF(KSW)401,402,401
402 WRITE(ITW,102){NX(I),I=1,4}
   KSW=1
401 WRITE(ITW,104){HEAD(I),I=1,4},SMSQ,NDF1,SMSQM
   TOTL = TOTL + SMSQ
   NDFRT = NDFRT + NDF1
   IF(INDI) 50,8,8
C   PRINT RESIDUAL AND/OR TITLE LINE
50 IDIF = KTDFR - NDFRT
   IF (IDIF) 51,52,51
51 SMSQ = TOT - TOTL
   SMSQM = SMSQ / IDIF
   WRITE (ITW,105) SMSQ,IDIF,SMSQM
52 WRITE (ITW,106) TOT,KTDFR
   RETURN
   END
// OUP
*STORE      WS UA REPR

```

```

RPRT 550 // FOR INPUT DATA SUBROUTINE FCTR 0
RPRT 560 *IOCS(CARD,1132PRINTER,DISK) FCTR 10
RPRT 570 *ONE WORD INTEGERS FCTR 20
RPRT 580 *NAME FCTR FCTR 30
RPRT 590 C INPUT DATA SUBROUTINE FCTR 40
RPRT 600 DEEINE EILE 606(500,65,U,IT1) FCTR 50
RPRT 610 DEFINE EILE 5(30,60,U,IT2) FCTR 60
RPRT 620 COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPR08,N,NF,CASES,NPAGE,INMD,IPRED, FCTR 70
RPRT 630 IICDM,IROT,NFRT,KX(1),MX(20),NCD(3), ISEQ,NCASE,KCNT,NX(9), FCTR 80
RPRT 640 ITRC,FLVB(4),KNN FCTR 90
RPRT 650 COMMON TITLE(18),VNAME(30),SUMY(30),SD(30),X(30),R(30,30) FCTR 100
RPRT 660 COMMON HIGH(30),HLOW(30),MF(50,3) ECTR 110
RPRT 670 101 FORMAT(6I2) FCTR 120
RPRT 680 102 FORMAT(I4,4X,18A4) FCTR 130
RPRT 690 103 FORMAT(3I12) FCTR 140
RPRT 700 104 FORMAT(20A4) FCTR 150
RPRT 710 105 FORMAT( 10X,18A4,5X,3HJ08,I7,5X,4HPAGE,I6//11X,19HNUMBER OF VARFCTR 160
RPRT 720 11ABLES,46X,I2/11X,10HINPUT TYPE , 55X,I2/11X,14HSEQUENCE CHECK 51XFCTR 170
RPRT 730 212/11X,19HVARIABLES ON CARD 1 46X,I2/11X,19HVARIABLES ON CARD 2 46XFCTR 180
RPRT 740 312/11X,19HVARIABLES ON CARD 3 46X,I2/11X,21HTRANSFORMATION SWITCH,ECTR 190
RPRT 750 444X,I2) FCTR 200
RPRT 760 106 FORMAT(11X,13HFACTOR SCORES 52X,I2/11X,24HNUMBER OF FACTORS OPTIONFCTR 210
RPRT 770 1 41X,I2/11X,37HNUMBER OF FACTORS OR PERCENT OF TRACE 28X,I2/11X,18FCTR 220
RPRT 780 2HCOMMUNALITY OPTION 47X,I2/11X,15HROTATION OPTION 50X,I2/11X,27HNUFCTR 230
RPRT 790 3MBER OF FACTORS TO ROTATE 38X,I2/11X,14HPOOLING OPTION FCTR 240
RPRT 800 451X,I2/11X,14HLATENT VECTORS 51X,I2/11X,23HUNROTATED FACTOR MATRIXFCTR 250
RPRT 810 5 42X,I2/11X,32HORTHOGONAL TRANSFORMATION MATRIX 33X,I2) FCTR 260
RPRT 820 107 FORMAT(11X,24HORTHOGONAL FACTOR MATRIX 41X,I2/11X,59HTRANSEORMATIOFCTR 270
1N MATRIX TO OBLIQUE REFERENCE VECTOR STRUCTURE 6X,I2/11X,41HOBLIQUFCTR 280
2E REFERENCE VECTOR STRUCTURE MATRIX 24X,I2/11X,44HCORRELATIONS AMOECTR 290
3NG OBLIQUE REFERENCE VECTORS 21X,I2/11X,39HOBLIQUE REFERENCE VECTNECTR 300
4R PATTERN MATRIX 26X,I2/11X,58HCORRELATIONS BETWEEN REEERENCE VECTFCTR 310
5ORS AND PRIMARY FACTORS 7X,I2) FCTR 320
108 FORMAT(///' AN ILLEGAL CHARACTER HAS BEEN ENCOUNTERED AT APPROXIMAFCTR 330
1TELY COLUMN',I3,' OF THE ABOVE DATA CARD.'/' CHANGE OR REMOVE CARDFCTR 340
2 AND PRESS START TO CONTINUE.') FCTR 350
109 EORMAT ( 11X,39HOBLIQUE PRIMARY FACTOR STRUCTECTR 360
1URE MATRIX 26X,I2/11X,42HCORRELATIONS AMONG OBLIQUE PRIMARY FACTORFCTR 370
2S 23X,I2/11X,37HOBLIQUE PRIMARY EACTOR PATTERN MATRIX 28X,I2/11X,3FCTR 380
36HEACTOR SCORE REGRESSION COEFFICIENTS 29X,I2) FCTR 390
110 FORMAT (11X,25HOUTPUT RAW CROSS PRODUCTS 40X,I2/11X 30HOUTPUT RESIFCTR 400
1DUAL CROSS PRODUCTS35X,I2/11X,28HOUTPUT VARIANCE - COVARIANCE37X,IFCTR 410
22/11X,18HOUTPUT CORRELATION 47X,I2) FCTR 420
111 FORMAT(/// 5X 4HCARD I10, I4,1X 30HIS OUT OF SEQUENCE. RERUN J08.ECTR 430
1) FCTR 440
112 FORMAT(///' AN ILLEGAL CHARACTER HAS BEEN ENCOUNTERED IN COLUMN', FCTR 450
113,' OF THE ABOVE FORMAT CARD.'/' CHANGE CARD AND RERUN J08.') FCTR 460
113 FORMAT(///' INVALID INPUT OPTION-J08 TERMINATED ') FCTR 470
C SUBROUTINE TO READ PARAMETER CARDS (FACTOR ANALYSIS) FCTR 480
NPAGE = 0 FCTR 490
READ(2,101) ICR,ICP,IPR,ITW,IT1,IT2 FCTR 500
IF(IPR)701,702,701 ECTR 510
702 ITW=3 ECTR 520
GO TO 703 FCTR 530
701 ITW=1 FCTR 540

```

```

703 READ(ICR,102) 1PRD8,TITLE
  READ(ICR,103) N,INMD,ISEQ,(NCD(I),I=1,3),MX(20),(MX(I),I=1,4),
  1IPRED,NF,KCNT,ICOM,1ROT,NFRT,NX(1),(MX(I),I=5,17)
  CALL FMAT(IPR,ITW)
  WRITE(ITW,105) TITLE,IPRO8,NPAGE,N,INMD,ISEQ,(NCD(I),I=1,3),
  1MX(20)
  WRITE(ITW,110) (MX(I),I=1,4)
  WRITE(ITW,106) 1PRED,NF,KCNT,ICOM,1ROT,NFRT,NX(1),(MX(I),I=5,7)
  WRITE(ITW,107) (MX(I),I=8,13)
  WRITE(ITW,109) (MX(I),I=14,17)
  READ(ICR,104) (VNAME(I),I=1,N)
  IF(INMD-1) 1,1,1001
1001 IF(INMD-4) 5,1002,1002
1002 WRITE(ITW,113)
  CALL EXIT
  DO 4 I=1,3
  CALL FMTRD(MF(1,I),IRR)
  CALL PRNT8
  IF(IRR) 2,3,2
  2 WRITE(ITW,112) .IRR....
  CALL EXIT
  3 IF(NCD(I+1)) 5,5,4
  4 CONTINUE
C  INITIALIZATION
  5 DO 8 I=1,N
    HIGH(I) = 0.
    HLOW(I) = 1.0E+36
    SUMY(I) = 0.
    SO(I)=0.0
    DO 8 J=1,N
      8 R(I,J) = 0.0
      KOUNT = 0
      CASES = 0.
      NCASE = 0
  9 IT1 = 1
  10 GO TO (11,41,51),INMD
C  CARD READER INPUT
  11 IST = 1
  I=1
  IF(NCD(1)) 12,12,13
  12 NCD(1) = N
  13 CALL DATRD(MF(1,1),IRR,ID,1,NC,I,X,-NCD(1),0,0)
  IF(IRR) 14,15,14
  14 CALL PRNT8
  WRITE(ITW,108) IRR
  PAUSE 10
  GO TO (13,18,18),I
  15 IF(ID) 150,16,16
  16 DO 22 I=2,3
    IF(NCD(I)) 23,23,17
  17 IST = NCD(I-1) + IST
  18 CALL DATRD(MF(1,I),IRR,1D1,1,NC1,1,X(IST),-NCD(I),0,0)
  IF(IRR) 14,19,14
  19 IF(ISEQ) 22,22,20
  20 IF(1D-ID1) 60,21,60

```

```

FCTR 550 21 IF(NC1-NC) 60,60,6
FCTR 560 6 ID = IDI
FCTR 570 NC = NC1
FCTR 580 22 CONTINUE
FCTR 590 GD TO 23
FCTR 600 60 WRITE(ITW,111) ID1,NC1
FCTR 610 CALL EXIT
FCTR 620 23 IF(MX(20)) 230,231,230
FCTR 630 230 CALL TRAN
FCTR 640 231 IF(INMD-2) 27,30,27
FCTR 650 27 WRITE(606*IT1) ID , (X(I),I=1,N)
FCTR 660 C COMPUTE CROSS PRODUCT MATRIX
FCTR 670 30 CASES = CASES + 1.
FCTR 680 NCASE = NCASE + 1
FCTR 690 DO 35 I = 1,N
FCTR 700 SUMY(I) = SUMY(I) + X(I)
FCTR 710 DO 35 J=I,N
FCTR 720 R(I,J) = R(I,J) + X(I)*X(J)
FCTR 730 35 R(J,I) = R(I,J)
FCTR 740 C DETERMINE HIGH AND LOW VALUES
FCTR 750 DO 39 I=1,N
FCTR 760 IF(X(I) - HIGH(I)) 37,37,36
FCTR 770 36 HIGH(I) = X(I)
FCTR 780 37 IF(X(I) - HLOW(I))38,39,39
FCTR 790 38 HLDW(I) = X(I)
FCTR 800 39 CONTINUE
FCTR 810 GO TO 10
FCTR 820 C READ DATA FROM DISK OR TAPE(360)
FCTR 830 41 READ(606*IT1) ID , (X(I),I=1,N)
FCTR 840 IF(ID) 43,43,23
FCTR 850 43 IT1=1
FCTR 860 GO TO 200
FCTR 870 C READ A MATRIX FROM CARDS
FCTR 880 51 IXOT=1ROT
FCTR 890 IRDT=NX(1)
FCTR 900 CALL MXRAD
FCTR 910 IRDT=IXOT
FCTR 920 IF(INMD-2)150,151,151
FCTR 930 150 WRITE(606*IT1) ID , (X(I),I=1,N)
FCTR 940 151 IT1 = 1
FCTR 950 200 IF(NCASE)400,400,300
FCTR 960 300 KNN=1
FCTR 970 CALL LINK(CDREL)
FCTR 980 400 CALL LINK(FCTR1)
FCTR 990 END
FCTR1000 // DUP
FCTR1010 *STORE WS UA FCTR
FCTR1020
FCTR1030
FCTR1040
FCTR1050
FCTR1060
FCTR1070
FCTR1080
FCTR1090

```

```

FCTR1100
FCTR1110
FCTR1120
FCTR1130
FCTR1140
FCTR1150
FCTR1160
FCTR1170
FCTR1180
FCTR1190
FCTR1200
FCTR1210
FCTR1220
FCTR1230
FCTR1240
FCTR1250
FCTR1260
FCTR1270
FCTR1280
FCTR1290
FCTR1300
FCTR1310
FCTR1320
FCTR1330
FCTR1340
FCTR1350
FCTR1360
FCTR1370
FCTR1380
FCTR1390
FCTR1400
FCTR1410
FCTR1420
FCTR1430
FCTR1440
FCTR1450
FCTR1460
FCTR1470
FCTR1480
FCTR1490
FCTR1500
FCTR1510
FCTR1520
FCTR1530
FCTR1540
FCTR1550
FCTR1560

```

```
// FOR FACTOR ANALYSIS SETUP PROGRAM
```

```
*ONE WORD INTEGERS
```

```
*IDCS(CARD,1132PRINTER,DISK)
```

```
*NAME FCTR1
```

```
C FACTOR ANALYSIS SETUP PROGRAM
```

```
  DEFINE FILE 606(500,65,U,IT1)
```

```
  DEFINE FILE 5(30,60,U,IT2)
```

```
  COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,ISCR,
```

```
  IICOM,IROT,NFRT,KX(1),MX(20),NCD1,NCD2,NCD3,ISEQ,NCASE,KCNT,NX(9),
```

```
  ITRC,FLV8(4),KNN
```

```
  COMMON TITLE(18),VNAME(30),SUMY(30),SD(30),DATA(30),R(30,30)
```

```
  COMMON Y(30),80000(30),80002
```

```
  9 IF(IICOM - 1) 30,10,20
```

```
C MAXIMUM ROW ELEMENT AS COMMUNALITY
```

```
  10 R(N,N) = 0.
```

```
    DO 12 I=1,N
```

```
      R(I,I) = ABS(R(1,N))
```

```
      DO 12 J=1,N
```

```
        IF(ABS(R(I,J)) - R(I,I)) 12,12,11
```

```
  11 R(I,I) = ABS(R(I,J))
```

```
  12 CONTINUE
```

```
    GO TO 30
```

```
C SQUARED MULTIPLE CORRELATION AS COMMUNALITY
```

```
  20 CALL INVR(R,DATA,N,IERR)
```

```
    DO 21 I=1,N
```

```
      R(I,I) = 1.-1./R(I,I)
```

```
      DO 21 J=1,N
```

```
  21 R(I,J) = R(J,I)
```

```
C COMPUTE TRACE OF THE MATRIX TO 8E FACTOREO
```

```
  30 TRC = 0.
```

```
    DO 31 I=1,N
```

```
  31 TRC = TRC + R(I,I)
```

```
C COMPUTE EIGENVALUES.
```

```
  CALL TRIDI
```

```
  CALL QR
```

```
  CALL LINK (FCTR2)
```

```
  END
```

```
// DUP
```

```
*STORE WS UA FCTR1
```

```
FCT1 0
```

```
FCT1 10
```

```
FCT1 20
```

```
FCT1 30
```

```
FCT1 40
```

```
FCT1 50
```

```
FCT1 60
```

```
FCT1 70
```

```
FCT1 80
```

```
FCT1 90
```

```
FCT1 100
```

```
FCT1 110
```

```
FCT1 120
```

```
FCT1 130
```

```
FCT1 140
```

```
FCT1 150
```

```
FCT1 160
```

```
FCT1 170
```

```
FCT1 180
```

```
FCT1 190
```

```
FCT1 200
```

```
FCT1 210
```

```
FCT1 220
```

```
FCT1 230
```

```
FCT1 240
```

```
FCT1 250
```

```
FCT1 260
```

```
FCT1 270
```

```
FCT1 280
```

```
FCT1 290
```

```
FCT1 300
```

```
FCT1 310
```

```
FCT1 320
```

```
FCT1 330
```

```
FCT1 340
```

```
FCT1 350
```

```
FCT1 360
```

```
FCT1 370
```

```
FCT1 380
```

```
// FOR
```

```
*ONE WORD INTEGERS
```

```
  SUBROUTINE INVR(R,X,NROW,IERR)
```

```
  DIMENSION R(30,30),X(30)
```

```
  IERR = 0
```

```
  DO 10 K=2,NROW
```

```
    M=K-1
```

```
    DO 3 KK=1,M
```

```
      X(KK) = 0.0
```

```
      DO 3 J=1,M
```

```
        IF(KK-J) 4,4,5
```

```
  5 X(KK) = X(KK) + R(J,KK)*R(J,K)
```

```
      GO TO 3
```

```
  4 X(KK) = X(KK) + R(KK,J)*R(J,K)
```

```
  3 CONTINUE
```

```
  ALPHA = R(K,K)
```

```
  DO 6 I=1,M
```

```
  6 ALPHA = ALPHA - X(I)*R(I,K)
```

```
  IF(ABS(ALPHA)-1.0E-8) 7,7,8
```

```
  7 IERR = 1
```

```
  GO TO 20
```

```
C CALCULATE LAST COLUMN OF NEXT INVERSE
```

```
  8 DO 9 I=1,M
```

```
    R(I,K) = -X(I)/ALPHA
```

```
C RECALCULATE PREVIOUS INVERSE
```

```
  DO 9 J=1,M
```

```
  9 R(I,J) = R(I,J) + (X(I)*X(J))/ALPHA
```

```
C CALCULATE R(K,K) ELEMENT OF NEXT INVERSE
```

```
  R(K,K) = 1.0/ALPHA
```

```
  10 CONTINUE
```

```
  20 RETURN
```

```
  ENO
```

```
// DUP
```

```
*STORE WS UA INVR
```

```
INVS 0
```

```
INVS 10
```

```
INVS 20
```

```
INVS 30
```

```
INVS 40
```

```
INVS 50
```

```
INVS 60
```

```
INVS 70
```

```
INVS 80
```

```
INVS 90
```

```
INVS 100
```

```
INVS 110
```

```
INVS 120
```

```
INVS 130
```

```
INVS 140
```

```
INVS 150
```

```
INVS 160
```

```
INVS 170
```

```
INVS 180
```

```
INVS 190
```

```
INVS 200
```

```
INVS 210
```

```
INVS 220
```

```
INVS 230
```

```
INVS 240
```

```
INVS 250
```

```
INVS 260
```

```
INVS 270
```

```
INVS 280
```

```
INVS 290
```

```
INVS 300
```

```
INVS 310
```

```
INVS 320
```

```
INVS 330
```

```

// FOR XMAX
*ONE WORD INTEGERS
  FUNCTION XMAX(A,B)
    XMAX = A
    IF(A-B)2,1,1
    2 XMAX = B
    1 RETURN
  END
// DUP
*STORE      WS UA XMAX

```

XMAX	0	// FOR TRANSFORM MATRIX TO TRIDIAGONAL FORM	TRID	0
XMAX	10	*ONE WORD INTEGERS	TRID	10
XMAX	20	C REDUCES A REAL SYMMETRIC N BY N MATRIX TO TRIDIAGONAL FORM USING	TRID	20
XMAX	30	C N - 2 ELEMENTARY ORTHOGONAL TRANSFORMATIONS. THE DIAGONAL	TRID	30
XMAX	40	C ELEMENTS AND THE SUBDIAGONAL ELEMENTS ARE STORED IN ARRAYS	TRID	40
XMAX	50	C ALPHA AND BETA RESPECTIVELY	TRID	50
XMAX	60	SUBROUTINE TRID1	TRID	60
XMAX	70	01MENSION GAM(30),V(30)	TRID	70
XMAX	80	COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,ISCR,	TRID	80
XMAX	90	ICOM,1ROT,NFRT,KX(1),MX(20),NX(15),TRC,FLV8(4),KNN	TRID	90
		COMMON TITLE(18),VNAME(30),SUMY(30),SD(30),P(30),R(30,30)	TRID	100
		COMMON ALPHA(30),BETA(30),ANORM	TRID	110
		ANORM=0.0	TRID	120
		ABS8=0.0	TRID	130
		L=N-2	TRID	140
		DO 4 K=1,L	TRID	150
		ALPHA(K) = R(K,K)	TRID	160
		SIGMA =0.0	TRID	170
		LL=K+1	TRID	180
		DO 5 I=LL,N	TRID	190
		5 SIGMA = R(I,K)*R(I,K)+SIGMA	TRID	200
		ABS8 = SQRT(SIGMA)	TRID	210
		T = ABS(ALPHA(K)) + ABS8	TRID	220
		ANORM = XMAX(ANORM,T+ABS8)	TRID	230
		A = R(K+1,K)	TRID	240
		B = SIGN(ABS8,-A)	TRID	250
		BETA(K) = B	TRID	260
		IF(SIGMA) 8,4,8	TRID	270
		8 GAMMA = 1.0 / (SIGMA - A*B)	TRID	280
		GAM(K)=GAMMA	TRID	290
		R(K+1,K)=A - B	TRID	300
		T=0.	TRID	310
		DO 13 I=LL,N	TRID	320
		P(I) = 0.	TRID	330
		DO 11 J=LL,I	TRID	340
		11 P(I) = P(I) + R(I,J)*R(J,K)	TRID	350
		IF(I-N)110,10,10	TRID	360
		110 L1 = I + 1	TRID	370
		DO 12 J=L1,N	TRID	380
		12 P(I) = P(I) + R(J,I)*R(J,K)	TRID	390
		10 P(I) =P(I)*GAMMA	TRID	400
		13 T = T + P(I) * R(I,K)	TRID	410
		T = .5*GAMMA *T	TRID	420
		DO 14 I=LL,N	TRID	430
		P(I) = P(I) - T*R(I,K)	TRID	440
		DO 14 J=LL,I	TRID	450
		14 R(I,J)=R(I,J) - R(I,K)*P(J) - R(J,K)*P(I)	TRID	460
		WRITE(5,K)(R(J,K),J=1,N)	TRID	470
		4 CONTINUE	TRID	480
		ALPHA(N-1)= R(N-1,N-1)	TRID	490
		BETA(N-1) = R(N,N-1)	TRID	500
		ALPHA(N) = R(N,N)	TRID	510
		BETA(N)=0	TRID	520
		T = ABS(BETA(N-1))	TRID	530
		ANORM=XMAX(ANORM , XMAX(ABS8+T+ABS(ALPHA(N-1)) , T+ABS(ALPHA(N))))	TRID	540

```

C      FORM TRANSFORMATION MATRIX BY APPLYING THE TRIDIAGONALIZING
C      ROTATIONS TO AN IDENTITY MATRIX.
      DO 402 I=1,N
      DO 403 J=1,N
403    R(I,J)=0.0
402    R(I,I)=1.0
      DO 409 I=1,L
      READ(5,1)(V(J),J=1,N)
      II = I + 1
      DO 409 J=2,N
      P(J)=0.0
      DO 408 K=II,N
408    P(J) = P(J) + R(J,K)*V(K)
      P(J) = P(J) * GAM(I)
      DO 409 K=II,N
409    R(J,K) = R(J,K) - P(J)*V(K)
      RETURN
      END
// DUP
*STORE      WS  UA  TRIDI

```

```

TRID 550
TRID 560
TRID 570
TRID 580
TRID 590
TRID 600
TRID 610
TRID 620
TRID 630
TRID 640
TRID 650
TRID 660
TRID 670
TRID 680
TRID 690
TRID 700
TRID 710
TRID 720
TRID 730
TRID 740

```

```

// FOR FIND EIGENVALUES OF TRIDIAGONAL MATRIX
*ONE WORD INTEGERS
C      FINDS THE EIGENVALUES OF A TRIDIAGONAL MATRIX BY THE QR METHOD
      SUBROUTINE QR
      DIMENSION A(30),B(30)
      COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,ISCR,
1ICOM,IROT,NFRT,KX(1),MX(20),NX(15),TRC,FLVB(4),KNN
      COMMON TITLE(18),VNAME(30),SUMY(30),SD(30),X(30),R(30,30)
      COMMON ALPHA(30),BETA(30),ANORM
      EPSQ = ANORM*ANORM*7.5E-14
C      SET INTERNAL ARRAYS A AND B TO ALPHA AND BETA**2 RESPECTIVELY.
      DO 542 I=1,N
      A(I)=ALPHA(I)
542    B(I) = BETA(I) * BETA(I)
      AMU = 0.0
      M = N
      1 IF(M-1)100,100,2
      2 I = M - 1
      K = I
      M1 = K
      IF(B(K)-EPSQ)3,3,4
      3 X(M) = A(M)
      AMU = 0.0
      M = K
      GO TO 1
      4 I = I - 1
      IF(I)7,7,5
      5 IF(B(I)-EPSQ)7,7,6
      6 K = I
      GO TO 4
      7 IF(K-M1)9,8,9
C      HANDLE 2 BY 2 BLOCK SEPARATELY.
      8 AMU = A(M1)*A(M) - B(M1)
      SQ1 = A(M1)+A(M)
      SQ2 = A(M1)-A(M)
      SQ2 = SQRT(SQ2*SQ2 + 4.0*B(M1))
      ALAMB = .5*(SQ1+SIGN(SQ2,SQ1))
      X(M1)=ALAMB
      X(M) = AMU/ALAMB
      AMU=0.0
      M = M - 2
      IF(M)101,101,1
C      SHORTCUT SINGLE QR ITERATION.
      9 ALAMB = 0.0
      IF(ABS(A(M)-AMU) - .5*ABS(A(M)))10,10,11
      10 ALAMB = A(M) + .5*SQRT(B(M1))
      11 AMU = A(M)
      SQ1=0.0
      SQ2=0.0
      U=0.0
      DO 20 I=K,M1
      GAMMA = A(I)-ALAMB-U
      IF(SQ1-1.0)12,13,12
      12 PQ = GAMMA*GAMMA/(1.0-SQ1)
      GO TO 15

```

```

QR00  0
QR00  10
QR00  20
QR00  30
QR00  40
QR00  50
QR00  60
QR00  70
QR00  80
QR00  90
QR00 100
QR00 110
QR00 120
QR00 130
QR00 140
QR00 150
QR00 160
QR00 170
QR00 180
QR00 190
QR00 200
QR00 210
QR00 220
QR00 230
QR00 240
QR00 250
QR00 260
QR00 270
QR00 280
QR00 290
QR00 300
QR00 310
QR00 320
QR00 330
QR00 340
QR00 350
QR00 360
QR00 370
QR00 380
QR00 390
QR00 400
QR00 410
QR00 420
QR00 430
QR00 440
QR00 450
QR00 460
QR00 470
QR00 480
QR00 490
QR00 500
QR00 510
QR00 520
QR00 530
QR00 540

```

```

13 PQ = 0.0
   IF(I-1)15,15,14
14 PQ = (1.0-SQ2)*B((-1)
15 T = PQ + B(I)
   IF(I-1)17,17,16
16 B(I-1)=SQ1*T
17 SQ2 = SQ1
   SQ1 = B(I)/T
   U = SQ1 * (GAMMA + A(I+1) - ALAMB)
   A(I) = GAMMA + U + ALAMB
20 CONTINUE
   GAMMA = A(M) -ALAMB-U
   IF(SQ1 - 1.0)21,22,21
21 B(M1)=SQ1*GAMMA*GAMMA/(1.0-SQ1)
   GO TO 23
22 B(M1) = SQ1*B(M1)*(1.0-SQ2)
23 A(M) = GAMMA + ALAMB
   GO TO 1
100 X(I)=A(I)
C  PLACE EIGENVALUES IN ORDER OF DESCENDING VALUE
101 DO IIO K=1,N
   XMN = -1000.
   DO 105 J=K,N
   IF(X(J) - XMN)105,105,103
103 XMN = X(J)
   JJ = J
105 CONTINUE
   X(JJ)=X(K)
   X(K)=XMN
110 CONTINUE
   RETURN
   END
// OUP
*STORE      WS  UA  QR

```

```

QR00 550 // FOR FACTOR ANALYSIS OUTPUT PROGRAM          FCT2  0
QR00 560 *IOCS(CAR0,1132PRINTER,DISK)                  FCT2 10
QR00 570 *ONE WORD INTEGERS                             FCT2 20
QR00 580 *NAME FCTR2                                     FCT2 30
QR00 590 C FACTOR ANALYSIS OUTPUT PROGRAM               FCT2 40
QR00 600   DEFINE FILE 6061500,65,U,IT1                 FCT2 50
QR00 610   DEFINE FILE 5130,60,U,IT2                     FCT2 60
QR00 620   DIMENSION Y(30)                               FCT2 70
QR00 630   COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMO,ISCR, FCT2 80
QR00 640   IICOM,IROT,NFRT,KX(I),MX(20),NC01,NC02,NC03,ISEQ,NCASE,KCNT,NX(9), FCT2 90
QR00 650   ITRC,FLVB(4),KNN                               FCT2 100
QR00 660   COMMON TITLE(18),VNAME(30),SUMY(30),SQ(30),  X(30),RI(30,30) FCT2 110
QR00 670   COMMON B1(30),B2I(30),B3                     FCT2 120
QR00 680 100 FORMAT(/2X13HJOB COMPLETED)                FCT2 130
QR00 690 101 FORMAT( 10X,18A4,5X,3HJOB,I7,5X,4HPAGE,I6) FCT2 140
QR00 700 102 FORMAT(/40X,5HTRACE,F15.4,/)                FCT2 150
QR00 710 103 FORMAT(38X,F15.4,15X,F15.4)                 FCT2 160
QR00 720 104 FORMAT(38X,20HCHARACTERISTIC ROOTS 10X,23HCUMUL. PERCENT OF TRACE FCT2 170
QR00 730 3 /, ' ' /)                                     FCT2 180
QR00 740 105 FORMAT( /10X13HCOMMUNALITIES/)              FCT2 190
QR00 750 106 FORMAT(2X,A4,5X,E15.7)                      FCT2 200
QR00 760 C DETERMINE NUMBER OF FACTORS TO COMPUTE        FCT2 210
QR00 770   DO 21 I=1,N                                    FCT2 220
QR00 780 21 Y(I) = 0.0                                     FCT2 230
QR00 790   IF(NF-2)32,33,40                               FCT2 240
QR00 800 32 NF = 0                                         FCT2 250
QR00 810   KCNT = 0                                       FCT2 260
QR00 820   GO TO 50                                       FCT2 270
QR00 830 33 NF = KCNT                                     FCT2 280
QR00 840   KCNT = 0                                       FCT2 290
QR00 850   GO TO 50                                       FCT2 300
QR00 860 40 NF = 0                                         FCT2 310
QR00 870 50 SUM = 0.                                       FCT2 320
QR00 880   PCNT = KCNT                                    FCT2 330
   DO 8 J=1,N                                              FCT2 340
   IF(X(J))1,1,2                                          FCT2 350
1   NF = J-1                                              FCT2 360
   GO TO 9                                                FCT2 370
2   IF(NF)4,4,3                                           FCT2 380
3   IF(NF-J)9,7,7                                         FCT2 390
4   IF(KCNT)5,5,6                                         FCT2 400
5   IF(X(J) - 1.0)1,7,7                                   FCT2 410
6   IF(YIJ) - PCNT)7,1,I                                  FCT2 420
7   SUM = SUM + X(J)                                       FCT2 430
8   Y(J) = SUM*100./TRC                                    FCT2 440
C  COMPUTE CHARACTERISTIC VECTOR FOR FIRST NF CHARACTERISTIC VALUES. FCT2 450
9  CALL VECTR                                             FCT2 460
C  OUTPUT CHARACTERISTIC VECTORS                          FCT2 470
   CALL PRNTIS(0,N,NF)                                    FCT2 480
C  COMPUTE FACTOR COEFFICIENTS                            FCT2 490
   DO 80 J=1,NF                                           FCT2 500
   SQTXX = SQRT(XIJ))                                     FCT2 510
   DO 80 I=1,N                                             FCT2 520
80  R(I,J) = R(I,J)*SQTXX                                  FCT2 530
C  OUTPUT CHARACTERISTIC VALUES                          FCT2 540

```

```

      DIFE = TRC - SUM
      NPAGE = NPAGE + 1
      CALL FMAT(IPR,ITW)
      IF(IPR) 81,81,82
81  WRITE(ITW,101) TITLE,IPROB,NPAGE
82  WRITE(ITW,102)TRC
      WRITE(ITW,104)
      DO 90 I=1,N
90  WRITE(ITW,103)          X(I),Y(I)
C   OUTPUT FACTOR MATRIX
      CALL PRNT(6,0,N,NF)
      WRITE(ITW,105)
      DO 52 I=1,N
      COM=0.
      DO 51 J=1,NF
51  COM=COM+R(I,J)**2
      WRITE(ITW,106)VNAME(I),COM
52  CONTINUE
      IF(IROT)16,17,16
16  CALL LINK(FCTR3)
      WRITE(ITW,100)
      CALL EXIT
      END
// DUP
*STORE      WS  UA  FCTR2

```

```

FCT2 550 // FOR SUBROUTINE TO OUTPUT RESULTS OF ROTATION ROUT 0
FCT2 560 *ONE WORD INTEGERS ROUT 10
FCT2 570 C SUBROUTINE TO OUTPUT RESULTS OF ROTATION ROUT 20
FCT2 580 SUBROUTINE RFOUT ROUT 30
FCT2 590 COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,ISCR, ROUT 40
FCT2 600 IICOM,IROT,NFRT,KX(1),MX(20),NX(15),TRC,FLVB(4),KNN ROUT 50
FCT2 610 COMMON TITLE(18),VNAME(30),SUMY(30),SO(30),H(30),A(30,10) ROUT 60
FCT2 620 COMMON B(10,10),E(10,10) ROUT 70
FCT2 630 C IF KX(1) = 0 ENTRY FROM VARMX ROUT 80
FCT2 640 C IF KX(1) = 1 ENTRY FROM PROMX ROUT 90
FCT2 650 IF(KX(1)) 1,1,10 ROUT 100
FCT2 660 1 CALL RPRNT(8,7,1,NFRT,NFRT) ROUT 110
FCT2 670 C OUTPUT ORTHOGONAL FACTOR MATRIX ROUT 120
FCT2 680 CALL RPRNT (A,8,0,N,NFRT) ROUT 130
FCT2 690 C SET B-MATRIX TO IDENTITY FOR FACTOR SCORES ROUT 140
FCT2 700 DO 4 I=1,NFRT ROUT 150
FCT2 710 DO 4 J=1,NFRT ROUT 160
FCT2 720 IF(I-J) 3,2,3 ROUT 170
FCT2 730 2 B(I,J) = 1.0 ROUT 180
FCT2 740 GO TO 4 ROUT 190
FCT2 750 3 B(I,J) = 0. ROUT 200
FCT2 760 4 CONTINUE ROUT 210
FCT2 770 GO TO 100 ROUT 220
FCT2 780 C OUTPUT OBLIQUE TRANSFORMATION MATRIX ROUT 230
FCT2 790 10 CALL RPRNT(8,9,1,NFRT,NFRT) ROUT 240
C OUTPUT CORRELATIONS AMONG OBLIQUE REFERENCE VECTORS ROUT 250
CALL RPRNT(E,11,1,NFRT,NFRT) ROUT 260
C OUTPUT OBLIQUE REFERENCE VECTOR STRUCTURE MATRIX ROUT 270
CALL RPRNT(A,10,0,N,NFRT) ROUT 280
C COMPUTE INVERSE OF REFERENCE VECTOR CORRELATIONS ROUT 290
CALL MATIN(E,NFRT) ROUT 300
C COMPUTE REFERENCE VECTOR PATTERN MATRIX (W) ROUT 310
DO 5 I=1,NFRT ROUT 320
5 WRITE(5*I) (A(J,I),J=1,N) ROUT 330
DO 12 I=1,N ROUT 340
DO 11 J = 1,NFRT ROUT 350
H(J) = 0.0 ROUT 360
DO 11 K = 1,NFRT ROUT 370
11 H(J) = H(J) + A(I,K) * E(K,J) ROUT 380
DO 12 J = 1,NFRT ROUT 390
12 A(I,J) = H(J) ROUT 400
C COMPUTE CDR. AMONG REFERENCE VECTORS AND PRIMARY FACTORS ROUT 420
DO 15 I = 1,NFRT ROUT 430
DO 15 J = 1,NFRT ROUT 440
IF (I-J) 14,13,14 ROUT 450
13 B(I,I) = 1. / SQRT(E(I,I)) ROUT 460
GO TO 15 ROUT 470
14 B(I,J) = 0.0 ROUT 480
15 CONTINUE ROUT 490
CALL RPRNT(8,13,1,NFRT,NFRT) ROUT 500
C COMPUTE CDR. AMONG PRIMARY FACTORS ROUT 510
DO 21 I = 1,NFRT ROUT 520
21 H(I) = B(I,I) ROUT 530
DO 20 I=1,NFRT ROUT 540

```

```

      DD 20 J = 1,NFRT
      IF (I-J) 16,17,16
16  B(I,J) = E(I,J) *H(I)*H(J)
      GO TO 20
17  B(I,I) = 1.0
20  CONTINUE
      CALL RPRNT(B,15,1,NFRT,NFRT)
C    COMPUTE PRIMARY FACTOR STRUCTURE MATRIX (S)
      DD 30 I = 1,NFRT
      DD 30 J = 1,N
30  A(J,I) = A(J,I) * H(I)
      CALL RPRNT (B,14,0,N,NFRT)
C    COMPUTE PRIMARY FACTOR PATTERN MATRIX
      DD 40 I = 1,NFRT
      READ(5*I) (A(J,I),J=I,N)
      DD 40 J = 1,N
40  A(J,I) = A(J,I)/H(I)
      CALL RPRNT(B,16,0,N,NFRT)
100 RETURN
      END
// DUP
*STORE      WS  UA  RFOUT

```

```

ROUT 550
ROUT 560
ROUT 570
ROUT 580
ROUT 590
ROUT 600
ROUT 610
ROUT 620
ROUT 630
ROUT 640
ROUT 650
ROUT 660
ROUT 670
ROUT 680
ROUT 690
ROUT 700
ROUT 710
ROUT 720
ROUT 730
ROUT 740
ROUT 750
ROUT 760

```

```

// FOR SUBROUTINE FOR OBLIQUE ROTATION (PROMAX)
*ONE WORD INTEGERS
C    SUBROUTINE FOR OBLIQUE ROTATION (PROMAX)
SUBROUTINE PROMX
COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,ISCR,
IICOM,IROT,NFRT,KX(I),MX(20),NX(I5),TRC,FLVB(4),KNN
COMMON TITLE(18),VNAME(30),SUMY(30),SD(30),H(30),A(30,10)
COMMON B(10,10),E(10,10),G(10)
C    COMPUTE A-TRANSPPOSE * A
21 DD 1 I=1,NFRT
    DD 1 J=1,NFRT
    B(I,J) = 0.
    DD 1 K=I,N
    1 B(I,J) = B(I,J) + A(K,I) * A(K,J)
    CALL MATIN(B,NFRT)
    DD 2 I=1,NFRT
    DD 2 J=1,NFRT
    E(I,J)=0.
    DD 2 K=I,N
    2 E(I,J)=E(I,J)+A(K,I)*SIGN((ABS(A(K,J)))**4),A(K,J))
    DD 8 I=1,NFRT
    DD 7 J=1,NFRT
    G(J) = 0.
    DD 7 K=1,NFRT
    7 G(J) = G(J) + B(I,K)*E(K,J)
    DD 8 J=1,NFRT
    8 B(I,J) = G(J)
    DD 10 J=I,NFRT
    T=0.
    DD 9 I=1,NFRT
    9 T = T + 8(I,J)**2
    T=SQRT(T)
    DD 10 I=I,NFRT
    10 B(I,J) = B(I,J)/T
C    APPLY TRANSFORMATION MATRIX TO FORM REFERENCE VECTOR STRUCTURE
C    MATRIX
    DD 12 I=1,N
    DD 11 J=1,NFRT
    G(J) = 0.
    DD 11 K=1,NFRT
    11 G(J) = G(J) + A(I,K)*8(K,J)
    DD 12 J=1,NFRT
    12 A(I,J) = G(J)
C    COMPUTE CORRELATIONS AMONG REFERENCE VECTORS
    DD 13 I=1,NFRT
    DD 13 J=1,NFRT
    E(I,J) = 0.
    DD 13 K=1,NFRT
    13 E(I,J) = E(I,J) + B(K,I)*B(K,J)
    KX(I) = 1
    RETURN
    END
// DUP
*STORE      WS  UA  PROMX

```

```

PRMX  0
PRMX 10
PRMX 20
PRMX 30
PRMX 40
PRMX 50
PRMX 60
PRMX 70
PRMX 80
PRMX 90
PRMX 100
PRMX 110
PRMX 120
PRMX 130
PRMX 140
PRMX 150
PRMX 160
PRMX 170
PRMX 180
PRMX 190
PRMX 200
PRMX 210
PRMX 220
PRMX 230
PRMX 240
PRMX 250
PRMX 260
PRMX 270
PRMX 280
PRMX 290
PRMX 300
PRMX 310
PRMX 320
PRMX 330
PRMX 340
PRMX 350
PRMX 360
PRMX 370
PRMX 380
PRMX 390
PRMX 400
PRMX 410
PRMX 420
PRMX 430
PRMX 440
PRMX 450
PRMX 460
PRMX 470
PRMX 480
PRMX 490
PRMX 500
PRMX 510
PRMX 520
PRMX 530

```

```

// FOR SUBROUTINE FOR ORTHOGONAL ROTATION (VARIMAX)
*ONE WORD INTEGERS
C SUBROUTINE FOR ORTHOGONAL ROTATION (VARIMAX)
  SUBROUTINE VARMX
    COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,ISCR,
    1ICOM,1ROT,NFRT,KX(1),MX(20),NX(15),TRC,FLVB(4),KNN
    COMMON TITLE(18),VNAME(30),SUMY(30),SD(30),H(30),A(30,10)
    COMMON B(10,10)
101 FORMAT( 10X,1BA4,5X,3HJOB,I7,5X,4HPAGE,I6)
102 FORMAT(26X13,1X2F15.8)
103 FORMAT(/42X,37HNORMAL VARIMAX CRITERION (NORMALIZED)//25X5HCYCLE6
1X9HCRITERION5X10HDIFFERENCE5X1BHEPSILON CRITERION=,F14.8)
C INITIALIZE VARIABLES.
  PREV=0.
  IF (NFRT) 50,50,52
50 IF (NF-10) 51,51,53
51 NFRT = NF
  GO TO 54
52 IF (NFRT-10) 54,54,53
53 NFRT = 10
54 EPS=0.00116
C FORM IDENTITY MATRIX FOR TRANSFORMATION
  DO 3 I=1,NFRT
    B(I,1) = 1.0
    DO 3 J=1,NFRT
      IF(I-J) 1,3,1
1 B(I,J) = 0.
3 CONTINUE
  LL = NFRT - 1
  NV = 0
  FN=N
  CONS = .7071068
C NORMALIZE INPUT MATRIX
  DO 5 I = 1,N
    H(I)=0.
    DO 4 J = 1,NFRT
      4 H(I) = H(I) + A(I,J)*A(I,J)
      H(I) = SQRT(H(I))
      DO 5 J = 1,NFRT
        5 A(I,J) = A(I,J) / H(I)
      NPAGE = NPAGE + 1
      CALL EMAT(IPR,ITW)
      IF(IPR) 501,501,6
501 WRITE(ITW,101) TITLE,IPROB,NPAGE
6 WRITE(ITW,103)EPS
C COMPUTE VARIANCE OF SQUARES FOR TO TEST CONVERGENCE.
61 TV = 0.
  NV=NV+1
  DO 8 J=1,NFRT
    SA = 0.
    SA2 = 0.
    DO 7 I=1,N
      ECCH = A(I,J) * A(I,J)
      SA = SA + ECCH
7 SA2 = SA2 + ECCH * ECCH
    V = (FN*SA2-SA*SA)/(FN*FN)
    TV = TV + V
    OIFFR = TV - PREV
    PREV = TV
    WRITE(ITW,102) NV,TV,DIFFR
    IF(INV - 5019,999,999
C IS THE VARIANCE ON THIS CYCLE EQUAL (APPROXIMATELY) TO LAST CYCLES?
9 IF(ABS(DIFFR)-.000001) 999,999,13
C BEGIN NEXT ITERATION CYCLE
13 DO 40 J=1,LL
  II=J+1
  DO 40 K=II,NFRT
C COMPUTE THE NUMERATOR AND DENOMINATOR OF THE TANGENT OF THETA.
  AA=0.0
  BB=0.0
  CC=0.0
  DD=0.0
  DO 15 I = 1,N
    XX=A(I,J)
    YY=A(I,K)
    UU = (XX + YY) * (XX - YY)
    VV = 2.0 * XX * YY
    CC = CC + ( UU + VV)*(UU - VV)
    DD = DD + UU * VV
    AA = AA + UU
15 BB = BB + VV
  T = 2.0 * (DD - AA * BB/FN)
  Z = CC - (AA * AA - BB*BB)/FN
  IF(T - Z)18,16,22
16 IF((T+Z)-EPS) 40,17,17
17 COST = .9807853
  SINT = .1950903
C THE SIN AND COSINE OF 11 DEGREES, 15 MINUTES ( 45/4 DEGREES )
  GO TO 26
18 TAN4T = ABS(T/Z)
  IF(TAN4T-EPS) 20,19,19
19 COS4T=1.0/SQRT(1.0+TAN4T**2)
  SIN4T=TAN4T*COS4T
  GO TO 25
20 IF(Z) 21,40,40
21 SINP=CONS
  COSP=CONS
  GO TO 31
C IF THE NUMERATOR IS MORE THAN THE DENOMINATOR, REVERSE THE TWO.
22 CTN4T = ABS(Z/T)
  IF(CTN4T - EPS) 24,23,23
C COMPUTE SUCCESSIVELY COS 2T, SIN 2T, COS T, SIN T.
23 SIN4T = 1.0/SQRT(1.0+CTN4T**2)
  COS4T = CTN4T*SIN4T
  GO TO 25
24 COS4T = 0.0
  SIN4T = 1.0
25 COS2T = CONS* SQRT(1. + COS4T)
  SIN2T=SIN4T/(2.0*COS2T)
  COST = CONS * SQRT(1. + COS2T)

```

```

      SINT=SIN2T/(2.0*COST)
26 IF(I) 28,28,27
27 COSP=COST
   SINT=SINT
   GO TO 29
C   IF DENOMINATOR IS NEGATIVE, SUBTRACT 45 DEGREES FROM THE ANGLE.
28 COSP = CONS * (COST + SINT)
   SINT = ABSICONS * (COST - SINT)
29 IF(I) 30,30,31
C   IF NUMERATOR WAS NEGATIVE, SUBTRACT 90 DEGREES FROM THE ANGLE.
30 SINT=-SINT
C   MULTIPLY THE TWO COLUMNS TO BE ROTATED BY THE MATRIX OF SINES AND
C   COSINES
31 DO 32 I=1,N
   AIJ = A(I,J) * COSP + A(I,K) * SINT
   AIK = -A(I,J)*SINT + A(I,K)*COSP
   A(I,J)=AIJ
32 A(I,K)=AIK
C   ROTATE THE CORRESPONDING COLUMNS OF THE IDENTITY MATRIX TO OBTAIN
C   THE TRANSFORMATION MATRIX.
   DO 33 I=1,NFRT
      COST = B(I,J) * COSP + B(I,K) * SINT
      SINT = B(I,K) * COSP - B(I,J) * SINT
      B(I,J) = COST
33 B(I,K) = SINT
40 CONTINUE
   GO TO 61
999 KX(1) = 0
   DO 2 I=1,N
   DO 2 J=1,NFRT
      2 A(I,J)=A(I,J)*H(I)
   RETURN
   END
// OUP
*STORE      WS  UA  VARMX

```

```

VRMX1100
VRMX1110
VRMX1120
VRMX1130
VRMX1140
VRMX1150
VRMX1160
VRMX1170
VRMX1180
VRMX1190
VRMX1200
VRMX1210
VRMX1220
VRMX1230
VRMX1240
VRMX1250
VRMX1260
VRMX1270
VRMX1280
VRMX1290
VRMX1300
VRMX1310
VRMX1320
VRMX1330
VRMX1340
VRMX1350
VRMX1360
VRMX1370
VRMX1380
VRMX1390
VRMX1400
VRMX1410
VRMX1420
VRMX1430
VRMX1440

```

```

// FOR FIND EIGENVECTORS OF TRIODIAGONAL MATRIX
*ONE WORD INTEGERS
C   FIND THE EIGENVECTORS OF THE TRIODIAGONAL MATRIX BY
C   THE METHOD OF J. H. WILKINSON
SUBROUTINE VECTR
  DIMENSION CONS(30),VECT(30)
  COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPRO8,N,NF,CASES,NPAGE,INMO,ISCR,
  IICOM,IROT,NFRT,KX(1),MX(20),NX(15),TRC,FLVB(4),KNN
  COMMON TITLE(18),VNAME(30),SUMY(30),SO(30),X(30),R(30,30)
  COMMON ALPHA(30),BETA(30),XX
  DO 500 K=1,NF
    XX = X(K)
C   INITIALIZE RIGHT SIDE OF EQUATIONS TO BE SOLVED TO ONES.
    DO 1 I=1,N
      1 CONS(I) = 1.0
      OLOH = 1.0
      DO 100 IJK = 1,10
        CALL COVEC(CONS,VECT)
      H = 0.0
      DO 2 I=1,N
        IF(A8S(H) - ABS(VECT(I)))11,2,2
      11 H = VECT(I)
      2 CONTINUE
      DO 4 I=1,N
        IF(A8S(CONS(I)/OLOH - VECT(I)/H) - 5.0E-2)4,45,45
      4 CONTINUE
      GO TO 200
C   IF RESULTS DO NOT CONVERGE, SET RIGHT-HAND SIDE TO LAST APPROX.
C   AND LOOP
      45 OLOH = H
      DO 100 J=1,N
        100 CONS(J) = VECT(J)
C   ONCE APPROX. SOLUTION HAS BEEN FOUND, REFINES IT TO FIVE PLACES.
      200 CONS(I)=CONS(I)-VECT(I)*(ALPHA(I)-XX) - VECT(2)*BETA(1)
      DO 201 I=2,N
        201 CONS(I) = CONS(I) - VECT(I-1) * BETA(I-1) - VECT(I+1) * BETA(I)
      1) - VECT(I) * (ALPHA(I)-XX)
      CALL COVEC(CONS,CONS)
      H = 0.0
      DO 212 I=1,N
        VECT(I) = VECT(I) + CONS(I)
        IF(A8S(H)-A8S(VECT(I)))211,212,212
      211 H = VECT(I)
      212 CONTINUE
C   REDUCE MAGNITUDE OF EIGENVECTOR TO PREVENT POSSIBLE OVERFLOWS.
      DO 3 I=1,N
        3 CONS(I) = VECT(I) / H
C   TRANSFORM EIGENVECTOR TO CORRESPONDING VECTOR OF ORIGINAL MATRIX
C   AND NORMALIZE
      H = 0.0
      DO 206 I=1,N
        VECT(I) = 0.0
      DO 205 J=1,N
        205 VECT(I) = VECT(I) + CONS(J)*R(I,J)
      206 H = H + VECT(I)*VECT(I)

```

```

VCTR 0
VCTR 10
VCTR 20
VCTR 30
VCTR 40
VCTR 50
VCTR 60
VCTR 70
VCTR 80
VCTR 90
VCTR 100
VCTR 110
VCTR 120
VCTR 130
VCTR 140
VCTR 150
VCTR 160
VCTR 170
VCTR 180
VCTR 190
VCTR 200
VCTR 210
VCTR 220
VCTR 230
VCTR 240
VCTR 250
VCTR 260
VCTR 270
VCTR 280
VCTR 290
VCTR 300
VCTR 310
VCTR 320
VCTR 330
VCTR 340
VCTR 350
VCTR 360
VCTR 370
VCTR 380
VCTR 390
VCTR 400
VCTR 410
VCTR 420
VCTR 430
VCTR 440
VCTR 450
VCTR 460
VCTR 470
VCTR 480
VCTR 490
VCTR 500
VCTR 510
VCTR 520
VCTR 530
VCTR 540

```

```

      H = SQRT(H)
      DO 210 I=1,N
210  VECT(I) = VECT(I) / H
      WRITE(5*K)(VECT(I),I=1,N)
500  CONTINUE
      DO 600 K=1,NF
      READ(5*K)(R(I,K),I=1,N)
600  CONTINUE
      RETURN
      END
// DUP
*STORE      WS UA  VECTR

```

```

VCTR 550 // FOR SOLVE SIMULTANEOUS TRIDIAGONAL EQUATIONS
VCTR 560 *ONE WORD INTEGERS
VCTR 570 C
VCTR 580 C PERFORM A SINGLE ITERATION OF WILKINSONS METHOD
VCTR 590 SUBROUTINE COVEC(CONS,VECT)
VCTR 600 C SOLVES THE SYSTEM OF EQUATIONS WHOSE GENERAL FORM IS-
VCTR 610 C  $BETA(I)*X(I-1) + (ALPHA(I)-XX)*X(I) + BETA(I)*X(I+1) = CONS(I)$ 
VCTR 620 C FOR  $X(1 - N)$ , WHERE  $BETA(0)=BETA(N+1)=0$ . AND XX IS AN EIGENVALUE
VCTR 630 C OF THE TRIDIAGONAL MATRIX DETERMINED BY THE ALPHAS AND BETAS.
VCTR 640 DIMENSION CONS(30),VECT(30)
VCTR 650 DIMENSION U(29),V(29),W(29)
VCTR 660 COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,ISCR,
1ICOM,IROT,NFRT,KX(1),MX(20),NX(15),TRC,FLVB(4),KNN
COMMON TITLE(18),VNAME(30),SUMY(30),SD(30),DATA(30),R(30,30)
COMMON ALPHA(30),BETA(30),XX
C = CONS(1)
P = ALPHA(1) - XX
Q = BETA(1)
I = 2
10 PP = BETA(I-I)
QQ = ALPHA(I) - XX
RR = BETA(I)
C SELECT MAXIMUM COEFFICIENT OF X(I) AS I TH PIVOT
4 IF(ABS(PP)-ABS(P))2,3,3
3 U(I-1)= CONS(I)/PP
V(I-1)=-QQ/PP
W(I-1)=-RR/PP
XP = P
P = XP * V(I-1) + Q
Q = XP * W(I-1)
C = C - XP * U ( I - 1 )
GO TO 5
2 U(I-1)= C/P
V(I-1) = -Q/P
W(I-1)=0.0
P = QQ + PP*V(I-1)
Q = RR
C = CONS(I) - PP*U(I-1)
5 I = I + 1
IF(I - N)10,11,12
11 PP = BETA(N-1)
QQ = ALPHA(N) - XX
RR = 0.0
GO TO 4
12 VECT(N) = C/P
C BACK SUBSTITUTION
14 DO 20 I=2,N
J = N+1-I
20 VECT(J) = U(J)+V(J)*VECT(J+1)+W(J)*VECT(J+2)
40 RETURN
END
// DUP
*STORE      WS UA  COVEC

```

```

CVEC 0
CVEC 10
CVEC 20
CVEC 30
CVEC 40
CVEC 50
CVEC 60
CVEC 70
CVEC 80
CVEC 90
CVEC 100
CVEC 110
CVEC 120
CVEC 130
CVEC 140
CVEC 150
CVEC 160
CVEC 170
CVEC 180
CVEC 190
CVEC 200
CVEC 210
CVEC 220
CVEC 230
CVEC 240
CVEC 250
CVEC 260
CVEC 270
CVEC 280
CVEC 290
CVEC 300
CVEC 310
CVEC 320
CVEC 330
CVEC 340
CVEC 350
CVEC 360
CVEC 370
CVEC 380
CVEC 390
CVEC 400
CVEC 410
CVEC 420
CVEC 430
CVEC 440
CVEC 450
CVEC 460
CVEC 470
CVEC 480
CVEC 490
CVEC 500
CVEC 510

```

```

// FOR ROTATIONS PACKAGE FOR FACTOR ANALYSIS
*ONE WORD INTEGERS
*IOCS(CARD,I132PRINTER,DISK)
*NAME FCTR3
C ROTATIONS PACKAGE FOR FACTOR ANALYSIS
  DEFINE FILE 606(500,65,U,IT1)
  DEFINE FILE 5(30,60,U,IT2)
  COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,ISCR,
  1ICOM,IROT,NFRT,KX(11),MX(20),NCD1,NCD2,NCD3,ISEQ,NCASE,NX(10),TRC,
  2FLVB(4),KNN
  COMMON TITLE(18),VNAME(30),SUMY(30),SD(30),X(30),R(30,10)
  COMMON B(10,10),E(10,10),G(10)
100 FORMAT(/2X13HJOB COMPLETED)
  4 IF(IROT-I) 9,5,5
  5 CALL VARMX
  CALL RFOUT
  IF(IROT-2) 7,6,6
  6 CALL PROMX
  CALL RFOUT
  7 IF(ISCR) 9,9,8
  8 CALL SCORE
  9 WRITE(ITW,I00)
  CALL EXIT
  END
// OUP
*STORE WS UA FCTR3

```

```

FCT3 0 // FOR MATRIX PRINT/PUNCH ROUTINE FOR ROTATION RPNT 0
FCT3 10 * ONE WORD INTEGERS RPNT 10
FCT3 20 C MATRIX PRINT/PUNCH ROUTINE FOR ROTATION RPNT 20
FCT3 30 SUBROUTINE RPRNT(B,MID,KODE,NR,NC) RPNT 30
FCT3 40 DIMENSION B(10,10) RPNT 40
FCT3 50 COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMD,KX(5), RPNT 50
FCT3 60 IMX(20),NX(15),FLVB(5),KNN RPNT 60
FCT3 70 COMMON TITLE(18),VNAME(30),SUMY(30),SD(30),DATA(30),R(30,10) RPNT 70
FCT3 80 101 FORMAT(5X44,4X8FI2.4) RPNT 80
FCT3 90 102 FORMAT(3X14,6X8FI2.4) RPNT 90
FCT3 100 103 FORMAT( 10X,18A4,5X,3HJOB,I7,5X,4HPAGE,I6) RPNT 100
FCT3 110 104 FORMAT(14,3I2,5E14.7) RPNT 110
FCT3 120 105 FORMAT(/103H READY THE PUNCH WITH BLANK CARDS AND PRESS START ON TRPNT 120
FCT3 130 1HE PUNCH AND CONSOLE. TURN CONSOLE SWITCH I5 ON.) RPNT 130
FCT3 140 106 FORMAT(1H ) RPNT 140
FCT3 150 201 FORMAT(3X ,8HVARIABLE,7X,8(I4,8X)/) RPNT 150
FCT3 160 327 FORMAT(/45X 32HORTHOGONAL TRANSFORMATION MATRIX ) RPNT 160
FCT3 170 328 FORMAT(/41X 33HORTHOGONAL FACTOR MATRIX(VARIMAX)) RPNT 170
FCT3 180 329 FORMAT(/45X 50HTRANSFORMATION TO OBLIQUE REFERENCE VECTOR STRCTR.) RPNT 180
FCT3 190 330 FORMAT(/45X 41HOBlique REFERENCE VECTOR STRUCTURE MATRIX ) RPNT 190
FCT3 200 331 FORMAT(/45X 44HCORRELATIONS AMONG OBLIQUE REFERENCE VECTORS) RPNT 200
FCT3 210 332 FORMAT(/45X 39HOBlique REFERENCE VECTOR PATTERN MATRIX ) RPNT 210
FCT3 220 333 FORMAT(/45X 48HCORR. BET. REFERENCE VECTORS AND PRIMARY FACTORS ) RPNT 220
FCT3 230 334 FORMAT(/45X 39HOBlique PRIMARY FACTOR STRUCTURE MATRIX ) RPNT 230
FCT3 240 335 FORMAT(/45X 35HCORR. AMONG OBLIQUE PRIMARY FACTORS ) RPNT 240
FCT3 250 336 FORMAT(/45X 31HOBlique PRIMARY FACTOR LOADINGS) RPNT 250
337 FORMAT(/45X 36HFACTOR SCORE REGRESSION COEFFICIENTS ) RPNT 260
IF(MX(MID)-1)I000,1,I00 RPNT 270
  1 I = 1 RPNT 280
  II = 8 RPNT 290
  ISW = MID-6 RPNT 300
  9 IF(NC-II) 10,11,11 RPNT 310
  10 II = NC RPNT 320
  11 NPAGE = NPAGE + 1 RPNT 330
  CALL FMAT(IPR,ITW) RPNT 340
  IF(IPR) 111,111,112 RPNT 350
111 WRITE(ITW,I03)TITLF,IPROB,NPAGE RPNT 360
112 GO TO (21,22,23,24,25,26,27,28,29,30,31),ISW RPNT 370
  21 WRITE(ITW,327) RPNT 380
  GO TO 32 RPNT 390
  22 WRITE(ITW,328) RPNT 400
  GO TO 32 RPNT 410
  23 WRITE(ITW,329) RPNT 420
  GO TO 32 RPNT 430
  24 WRITE(ITW,330) RPNT 440
  GO TO 32 RPNT 450
  25 WRITE(ITW,331) RPNT 460
  GO TO 32 RPNT 470
  26 WRITE(ITW,332) RPNT 480
  GO TO 32 RPNT 490
  27 WRITE(ITW,333) RPNT 500
  GO TO 32 RPNT 510
  28 WRITE(ITW,334) RPNT 520
  GO TO 32 RPNT 530
  29 WRITE(ITW,335) RPNT 540

```

```

      GO TO 32
30  WRITE(ITW,336)
      GO TO 32
31  WRITE(ITW,337)
32  WRITE(ITW,201)(J,J=I,II)
      DO 35 K=1,NR
          IF(KODE) 34,33,34
33  WRITE(ITW,101) VNAME(K),(R(K,J),J=I,II)
      GO TO 35
34  WRITE(ITW,102) K,(8(K,J),J=1,II)
35  CONTINUE
      IF(NC-II) 36,1000,36
36  I = I+8
      II = II + 8
      GO TO 9
C    PUNCH ROUTINE
100  I = 1
      II = 5
      READ(ICR,106)
      CALL DATSW(15,JIG)
      IF(JIG-2)151,3,3
      3  WRITE(ITW,105)
          PAUSE
151  IF(NC-II) 152,153,153
152  II = NC
153  DO 156 K = 1,NR
          IF(KODE) 154,154,155
154  WRITE(ICP,104)IPROB,MID,I ,K,(R(K,J),J=I,II)
      GO TO 156
155  WRITE(ICP,104)IPROB,MID,I ,K,(8(K,J),J=I,II)
156  CONTINUE
      IF(NC-II)157,158,157
157  I = I + 5
      II = II + 5
      GO TO 151
158  IF(MX(MID)-2) 1000,1,1000
1000 RETURN
      END
// DUP
*STORE      WS  UA  RPRNT

```

```

RPNT 550 // FOR SUBROUTINE TO INVERT A MATRIX
RPNT 560 *ONE WORD INTEGERS
RPNT 570 C SUBROUTINE TO INVERT A MATRIX
RPNT 580 SUBROUTINE MATIN(A,N)
RPNT 590 DIMENSION IPIV(10),A(10,10),INDEX(10,2),PIVOT(10)
RPNT 600 DO 20 J = 1,N
RPNT 610 20 IPIV(J) = 0
RPNT 620 DO 550 I = 1,N
RPNT 630 AMAX = 0.0
RPNT 640 DO 105 J = 1,N
RPNT 650 IF(IPIV(J) - 1)60,105,60
RPNT 660 60 DO 100 K = 1,N
RPNT 670 IF(IPIV(K) - 1)80,100,740
RPNT 680 80 IF(ABS(AMAX) - ABS(A(J,K)))85,100,100
RPNT 690 85 IROW = J
RPNT 700 ICOLM = K
RPNT 710 AMAX = A(J,K)
RPNT 720 100 CONTINUE
RPNT 730 105 CONTINUE
RPNT 740 IPIV(ICOLM) = IPIV(ICOLM) + 1
RPNT 750 IF(IROW - ICOLM)150,260,150
RPNT 760 150 DO 200 L = 1,N
RPNT 770 SWAP = A(IROW,L)
RPNT 780 A(IROW,L) = A(ICOLM,L)
RPNT 790 200 A(ICOLM,L) = SWAP
RPNT 800 260 INDEX(1,1) = IROW
RPNT 810 INDEX(1,2) = ICOLM
RPNT 820 PIVOT(1) = A(ICOLM,ICOLM)
RPNT 830 A(ICOLM,ICOLM) = 1.0
RPNT 840 DO 350 L = 1,N
RPNT 850 A(ICOLM,L) = A(ICOLM,L) / PIVOT(1)
RPNT 860 DO 550 LL = 1,N
RPNT 870 IF(LL - ICOLM)400,550,400
RPNT 880 400 T = A(LL,ICOLM)
RPNT 890 A(LL,ICOLM) = 0.0
RPNT 900 DO 450 I = 1,N
RPNT 910 450 A(LL,I) = A(LL,I) - A(ICOLM,I) * T
RPNT 920 550 CONTINUE
RPNT 930 DO 710 I = 1,N
RPNT 940 L = N + 1 - I
      IF(INDEX(L,1) - INDEX(L,2))630,710,630
630 JROW = INDEX(L,1)
      JCOLM = INDEX(L,2)
      DO 705 K = 1,N
          SWAP = A(K,JROW)
          A(K,JROW) = A(K,JCOLM)
705 A(K,JCOLM) = SWAP
710 CONTINUE
740 RETURN
      END
// DUP
*STORE      WS  UA  MATIN

```

```

MATN  C
MATN 10
MATN 20
MATN 30
MATN 40
MATN 50
MATN 60
MATN 70
MATN 80
MATN 90
MATN 100
MATN 110
MATN 120
MATN 130
MATN 140
MATN 150
MATN 160
MATN 170
MATN 180
MATN 190
MATN 200
MATN 210
MATN 220
MATN 230
MATN 240
MATN 250
MATN 260
MATN 270
MATN 280
MATN 290
MATN 300
MATN 310
MATN 320
MATN 330
MATN 340
MATN 350
MATN 360
MATN 370
MATN 380
MATN 390
MATN 400
MATN 410
MATN 420
MATN 430
MATN 440
MATN 450
MATN 460
MATN 470
MATN 480
MATN 490
MATN 500
MATN 510

```

// FOR SUBROUTINE TO COMPUTE FACTOR SCORES	SCOR 0	18 READ(606,IT1)IO, (X(I),I=1,N)	SCOR 550
*ONE WORD INTEGERS	SCOR 10	IF(IO) 50,50,19	SCOR 560
C SUBROUTINE TO COMPUTE FACTOR SCORES	SCOR 20	19 DO 20 I=1,N	SCOR 570
SUBROUTINE SCORE	SCOR 30	20 X(I) = (X(I)-SUMY(I))/SO(I)	SCOR 580
COMMON ICR,ICP,IPR,ITW,IT1,IT2,IPROB,N,NF,CASES,NPAGE,INMO,ISCR,	SCOR 40	DO 21 J=1,NFRT	SCOR 590
1ICOM,IROT,NFRT,KX(1),MX(20),NX(15),TRC,FLVB(4),KNN	SCOR 50	Z(J) = 0.	SCOR 600
COMMON TITLE(I8),VNAME(30),SUMY(30),SO(30),X(30),A(30,10)	SCOR 60	DO 21 I=1,N	SCOR 610
COMMON B(10,10),Z(10)	SCOR 70	21 Z(J) = Z(J) + A(I,J)*X(I)	SCOR 620
101 FORMAT( 10X,1BA4,5X,3HJOB,I7,5X4HPAGE,I6)	SCOR 80	C OUTPUT FACTOR SCORES	SCOR 630
102 FORMAT(//45X,13HFACTOR SCORES,// 2X14HIDENTIFICATION2X,5I14, /18X5I	SCOR 90	IF(LINES-79) 26,25,25	SCOR 640
114//)	SCOR 100	25 NPAGE = NPAGE + 1	SCOR 650
103 FORMAT( 1X15,1X,I7,6X,5E14.5/20X,5E14.5)	SCOR 110	LINES = 0	SCOR 660
104 FORMAT(I4,I2,5E14.7/5E14.7)	SCOR 120	CALL FMAT(IPR,ITW)	SCOR 670
105 FORMAT(IH )	SCOR 130	IF(IPR) 251,251,26	SCOR 680
106 FORMAT(/103H READY THE PUNCH WITH BLANK CARDS AND PRESS START ON T	SCOR 140	251 WRITE(ITW,101) TITLE,IPROB,NPAGE	SCOR 690
THE PUNCH AND CONSOLE. TURN CONSOLE SWITCH 15 ON.)	SCOR 150	26 IF(LINES)41,42,41	SCOR 700
I25=25	SCOR 160	42 WRITE(ITW,102)(K,K=1,NFRT)	SCOR 710
C COMPUTE COMMUNALITIES	SCOR 170	41 LINES = 2+LINES + (NFRT-1)/10	SCOR 720
DO 1 I=1,N	SCOR 180	II = II + 1	SCOR 730
X(I) = 0.	SCOR 190	30 WRITE(ITW,103) II, IO, (Z(J),J=1,NFRT)	SCOR 740
DO 10 J=1,NFRT	SCOR 200	IF(ISCR-2)18,301,18	SCOR 750
10 X(I) = X(I) - A(I,J)**2	SCOR 210	301 WRITE(ICP,104)II,I25,(Z(J),J=1,NFRT)	SCOR 760
1 X(I)=1.+X(I)	SCOR 220	GO TO 18	SCOR 770
DO B J=1,N	SCOR 230	50 RETURN	SCOR 780
DO 11 I=1,NFRT	SCOR 240	END	SCOR 790
11 Z(I)=A(I,I)/X(I)	SCOR 250	// OUP	SCOR 800
B WRITE(5,J) (Z(K),K=1,NFRT)	SCOR 260	*STORE WS UA SCORE	SCOR 810
CALL MATIN(B,NFRT)	SCOR 270		
C A-TRANSPPOSE * UNIQUENESS * A, ZOOED TO PHI	SCOR 280		
DO 13 I=1,NFRT	SCOR 290		
DO 12 J=1,NFRT	SCOR 300		
Z(J) = 0.0	SCOR 310		
DO 12 K=1,N	SCOR 320		
12 Z(J) = Z(J) + A(K,J)*A(K,I)/X(K)	SCOR 330		
DO 13 J=1,NFRT	SCOR 340		
13 B(I,J) = B(1,J) + Z(J)	SCOR 350		
CALL MATIN(B,NFRT)	SCOR 360		
C COMPUTE FACTOR SCORE COEFFICIENTS	SCOR 370		
DO 14 I=1,N	SCOR 380		
READ(5,I) (Z(L),L=1,NFRT)	SCOR 390		
DO 14 J=1,NFRT	SCOR 400		
A(I,J) = 0.	SCOR 410		
DO 14 K=1,NFRT	SCOR 420		
14 A(I,J) = A(I,J) + B(J,K)*Z(K)	SCOR 430		
CALL RPRNT(B,I7,0,N,NFRT)	SCOR 440		
C COMPUTE FACTOR SCORES	SCOR 450		
IT1 = 1	SCOR 460		
IF(ISCR-2)302,303,302	SCOR 470		
303 READ(ICR,I05)	SCOR 480		
CALL OATSW(15,JIG)	SCOR 490		
IF(JIG-2)302,3,3	SCOR 500		
3 WRITE(ITW,106)	SCOR 510		
PAUSE	SCOR 520		
302 LINES = 79	SCOR 530		
II=0	SCOR 540		

```

// FOR
*ONE WORD INTEGERS
  SUBROUTINE FMAT(IPR,ITW)
    IF (IPR) 1,1,2
      1 WRITE(ITW,100)
      GO TO 3
      2 WRITE(ITW,101)
      3 RETURN
    100 FORMAT(1H1)
    101 FORMAT(///' '///)
  END
// DUP
*STORE      WS UA FMAT
1F0031217

```

```

FMAT  0
FMAT 10
FMAT 20
FMAT 30
FMAT 40
FMAT 50
FMAT 60
FMAT 70
FMAT 80
FMAT 90
FMAT 100
FMAT 110
FMAT 120

```

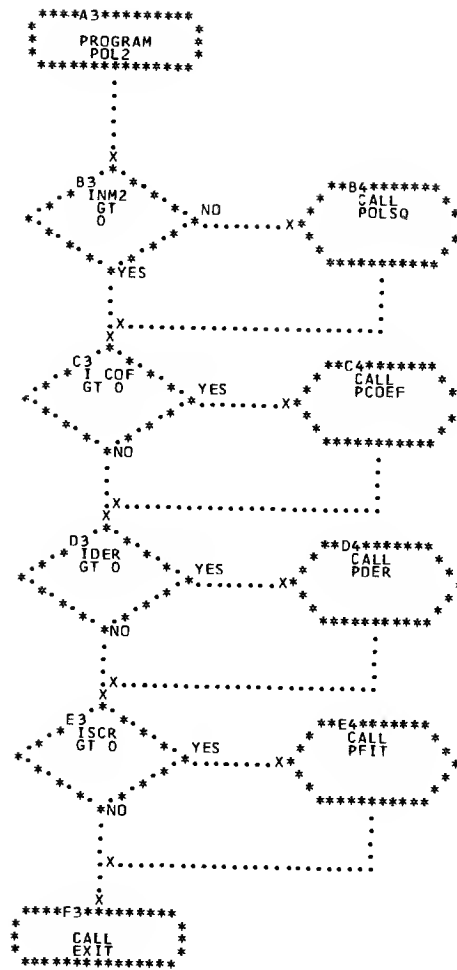
CHART. DA

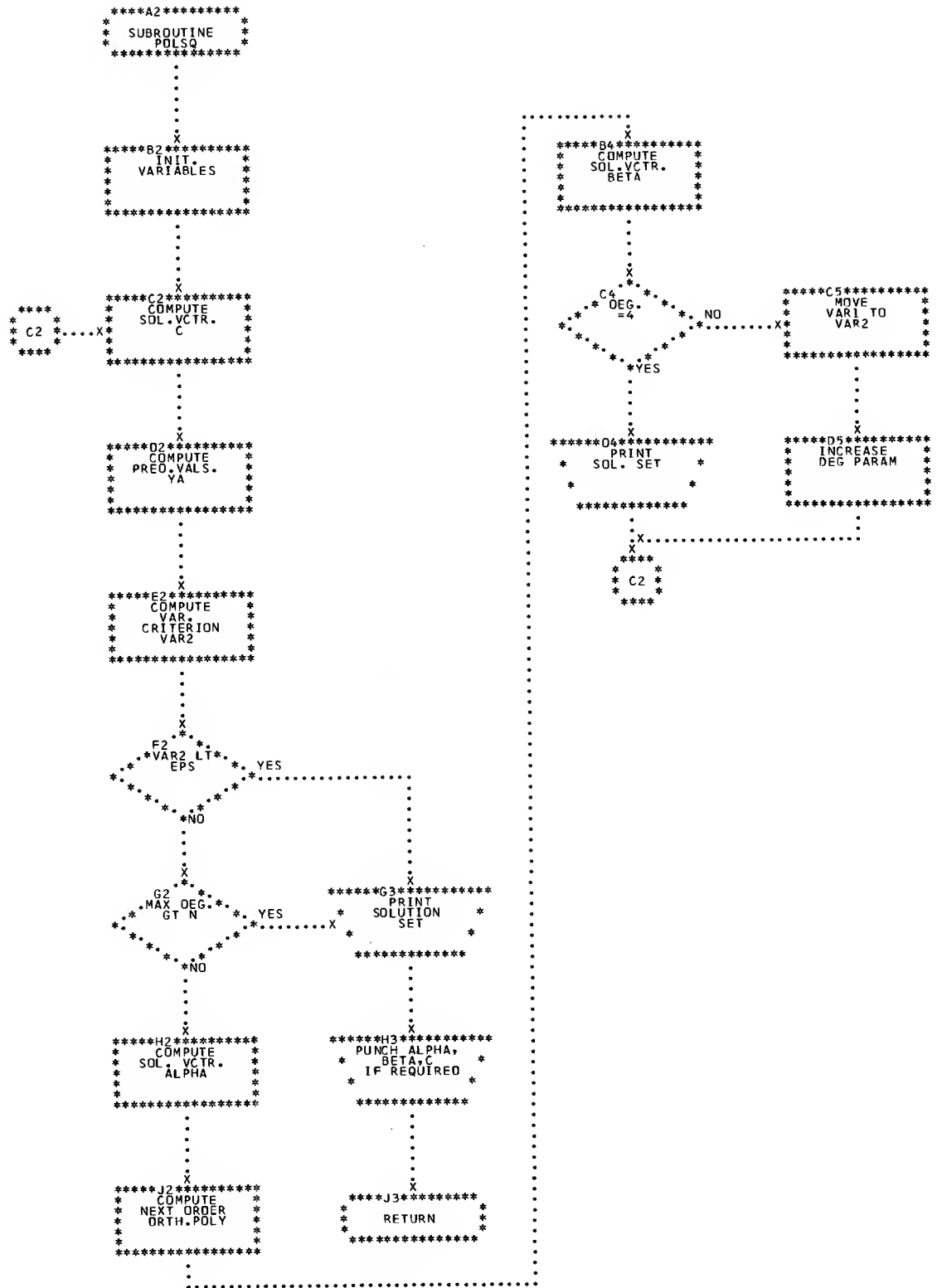






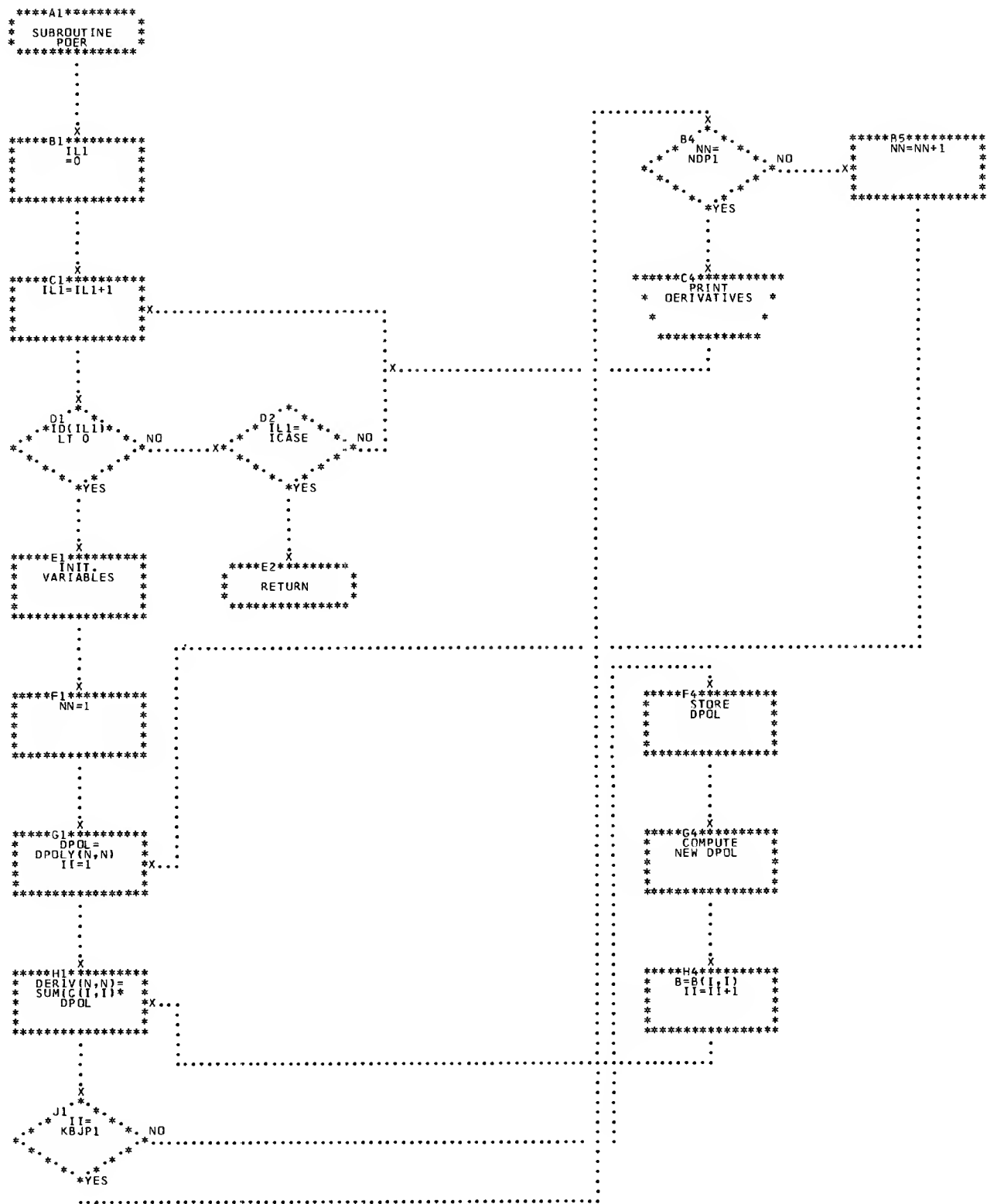
[illegible]

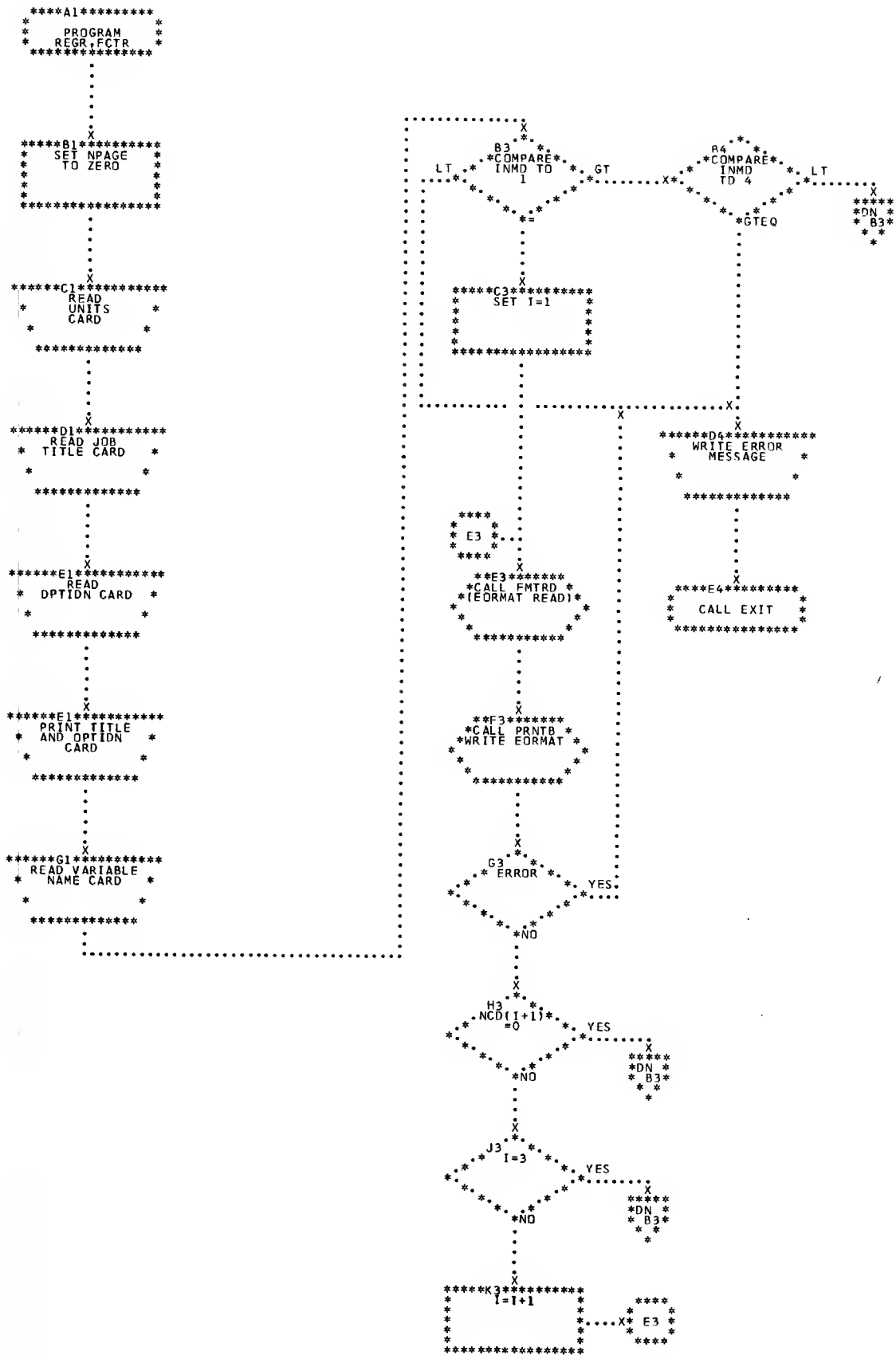


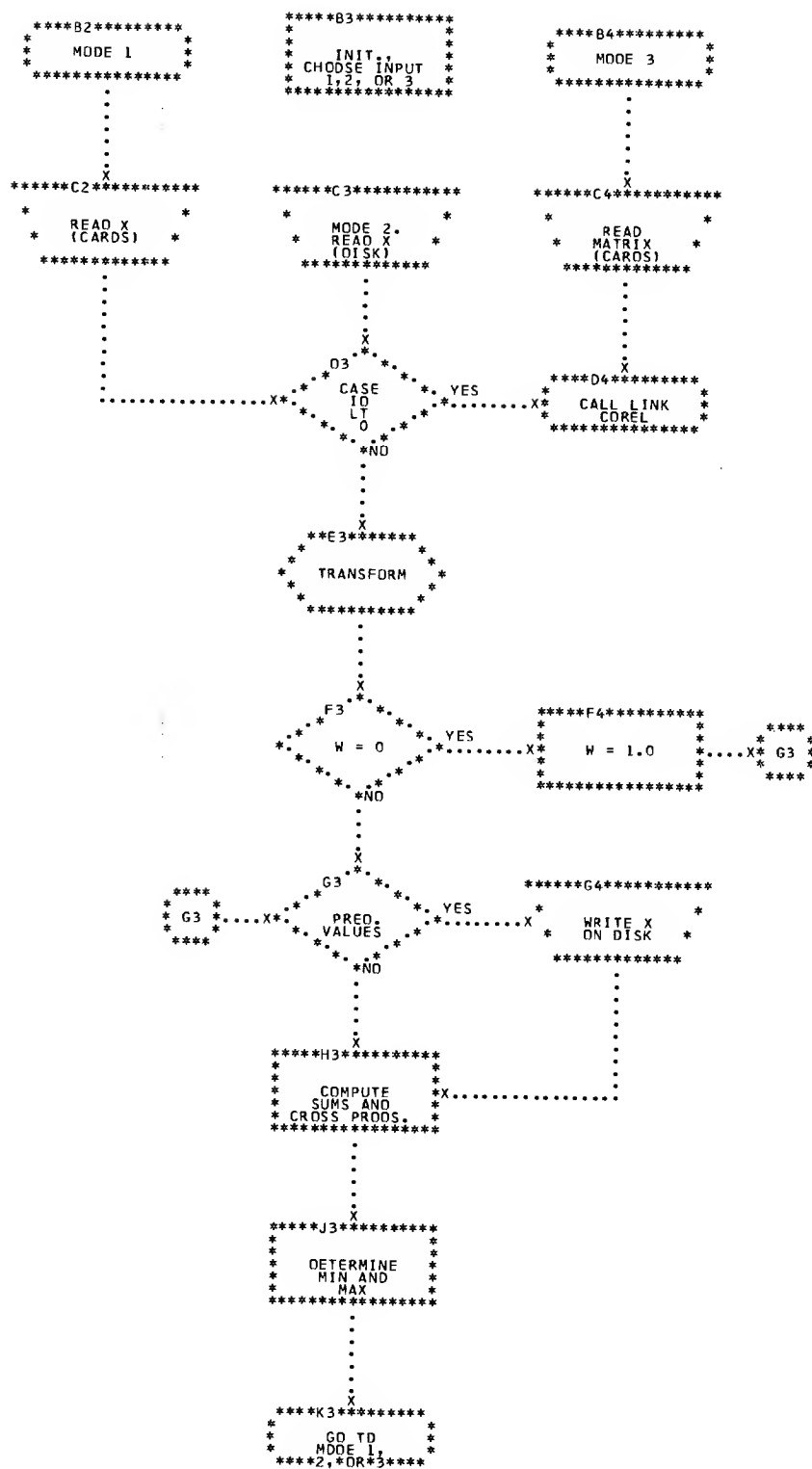


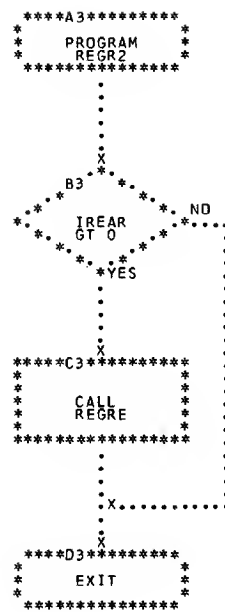


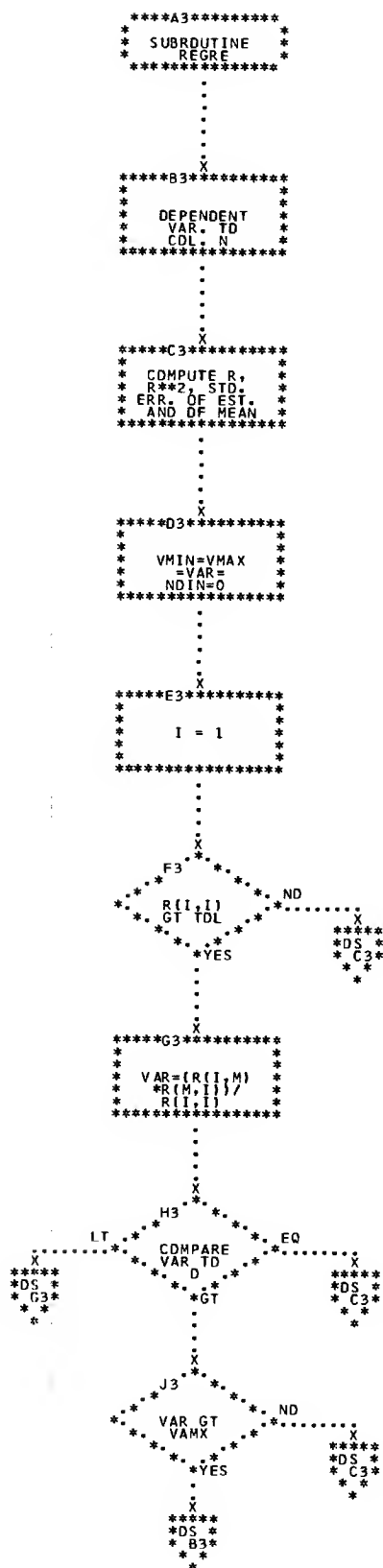




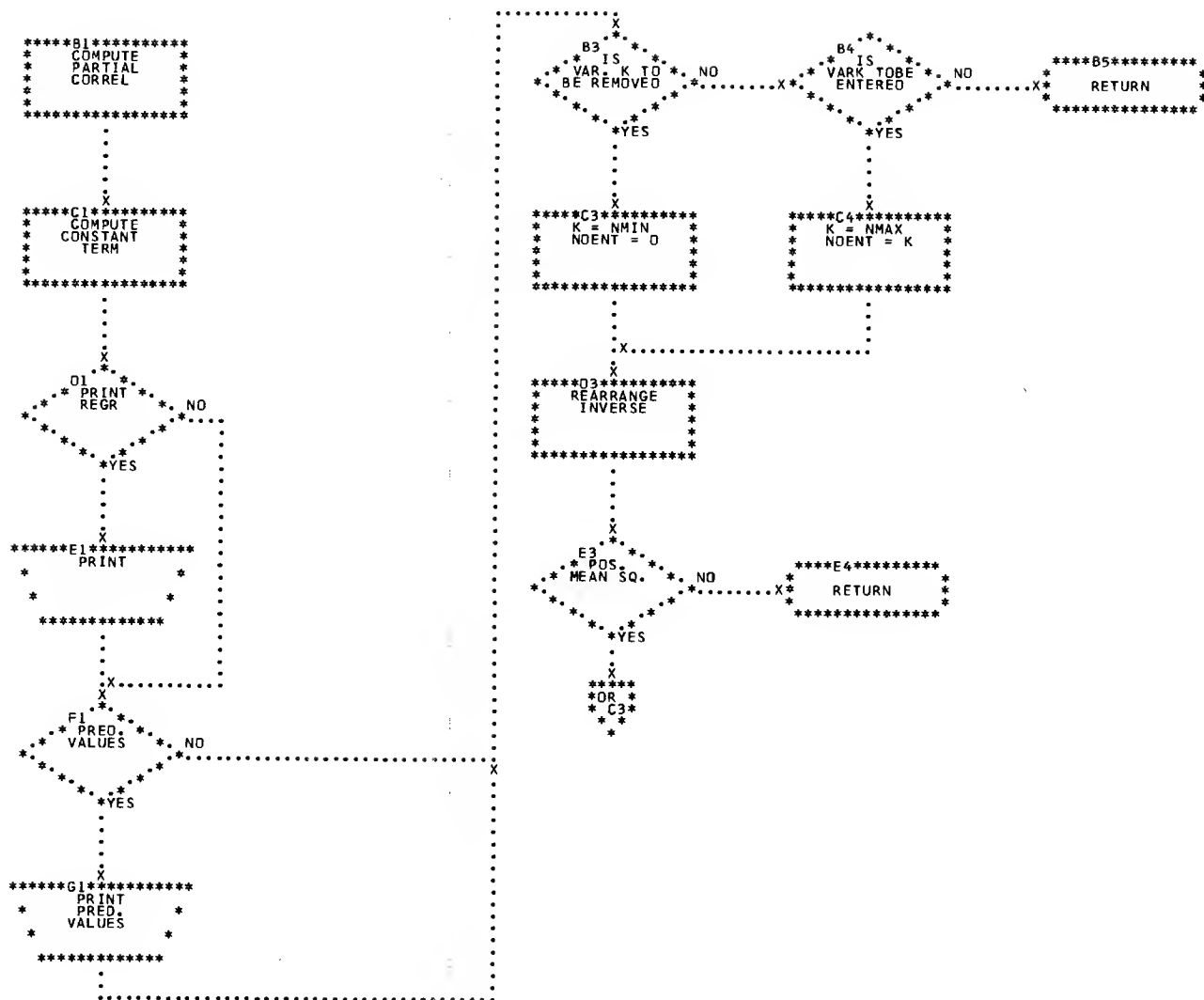


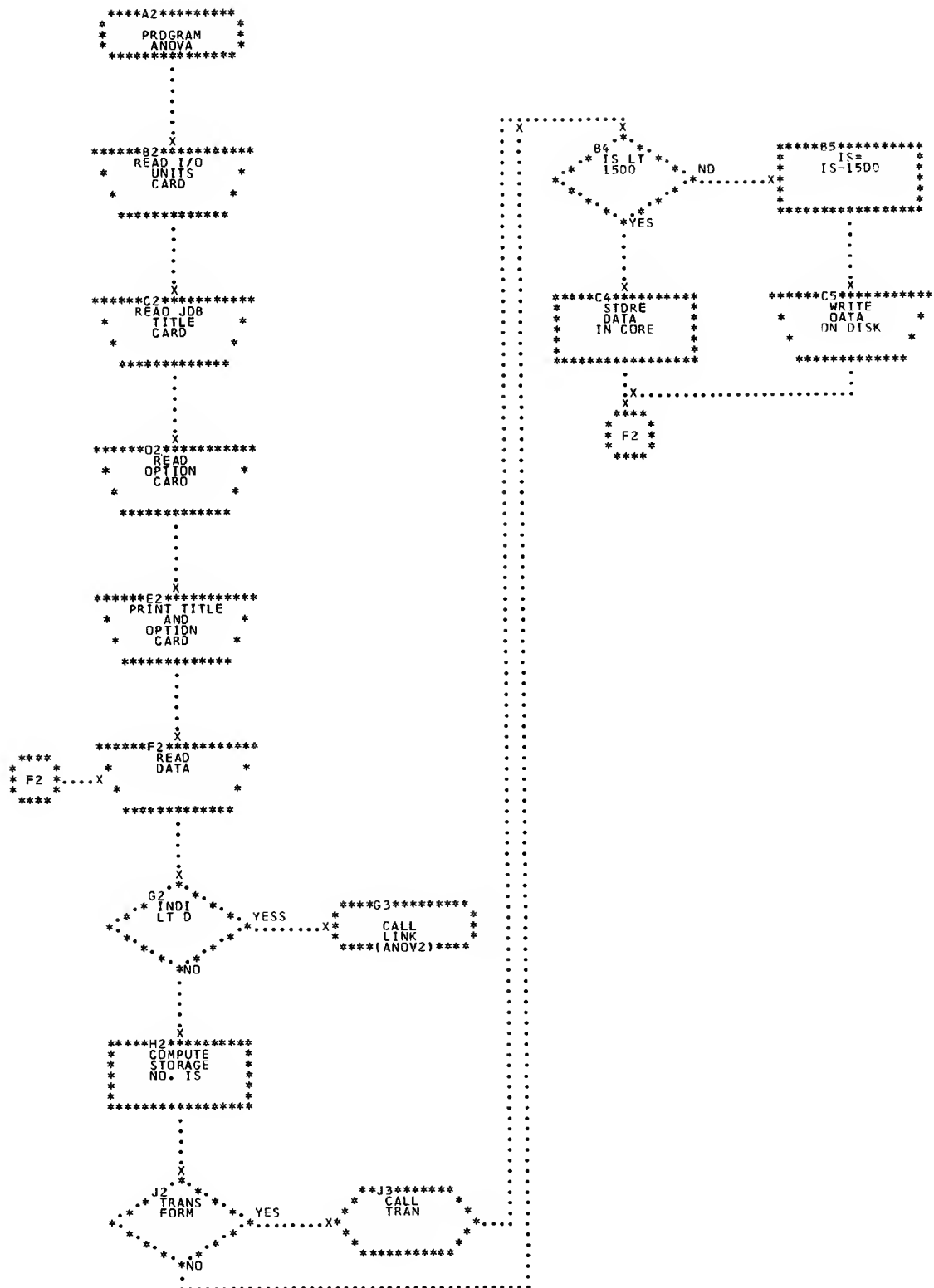






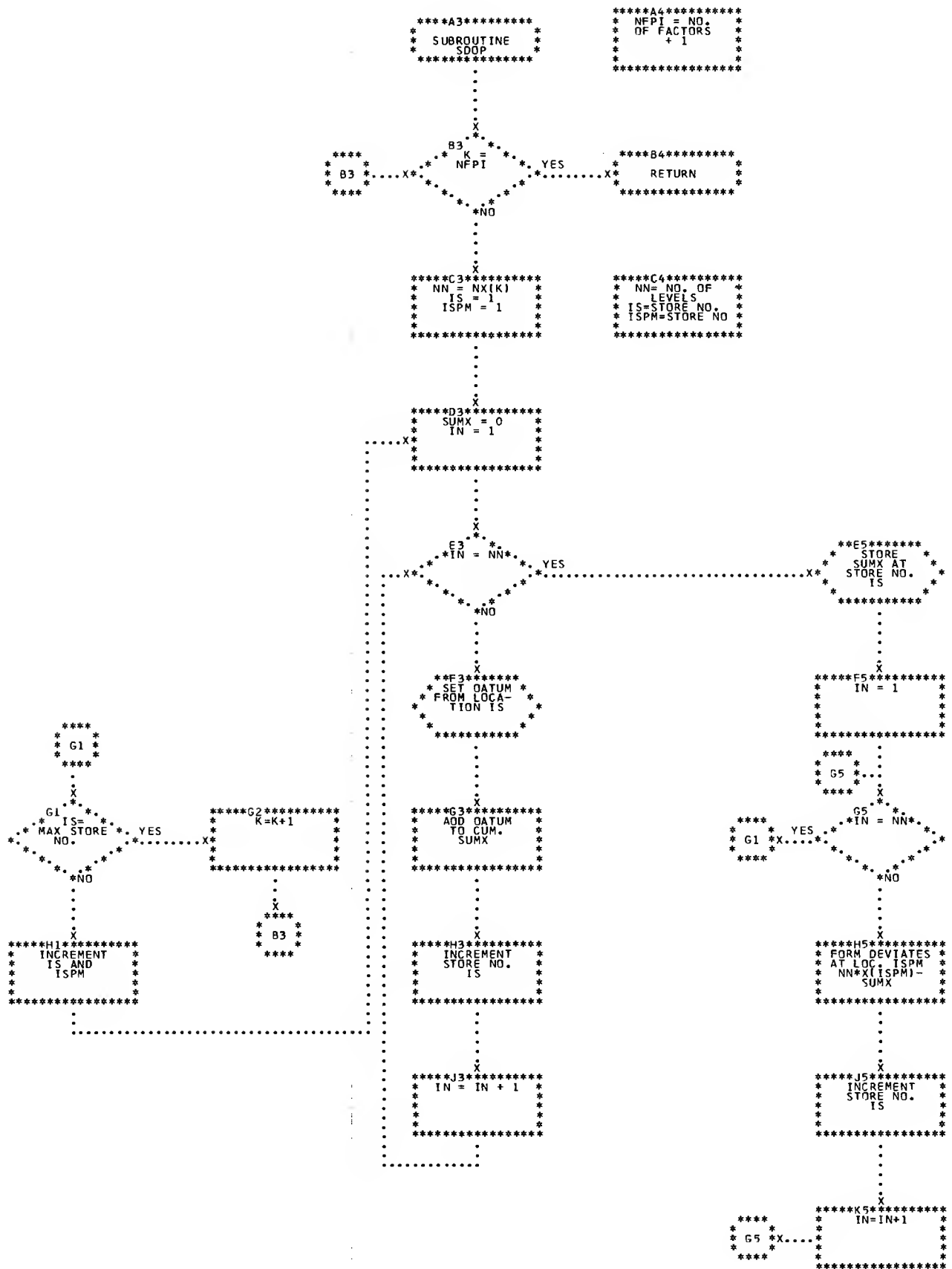




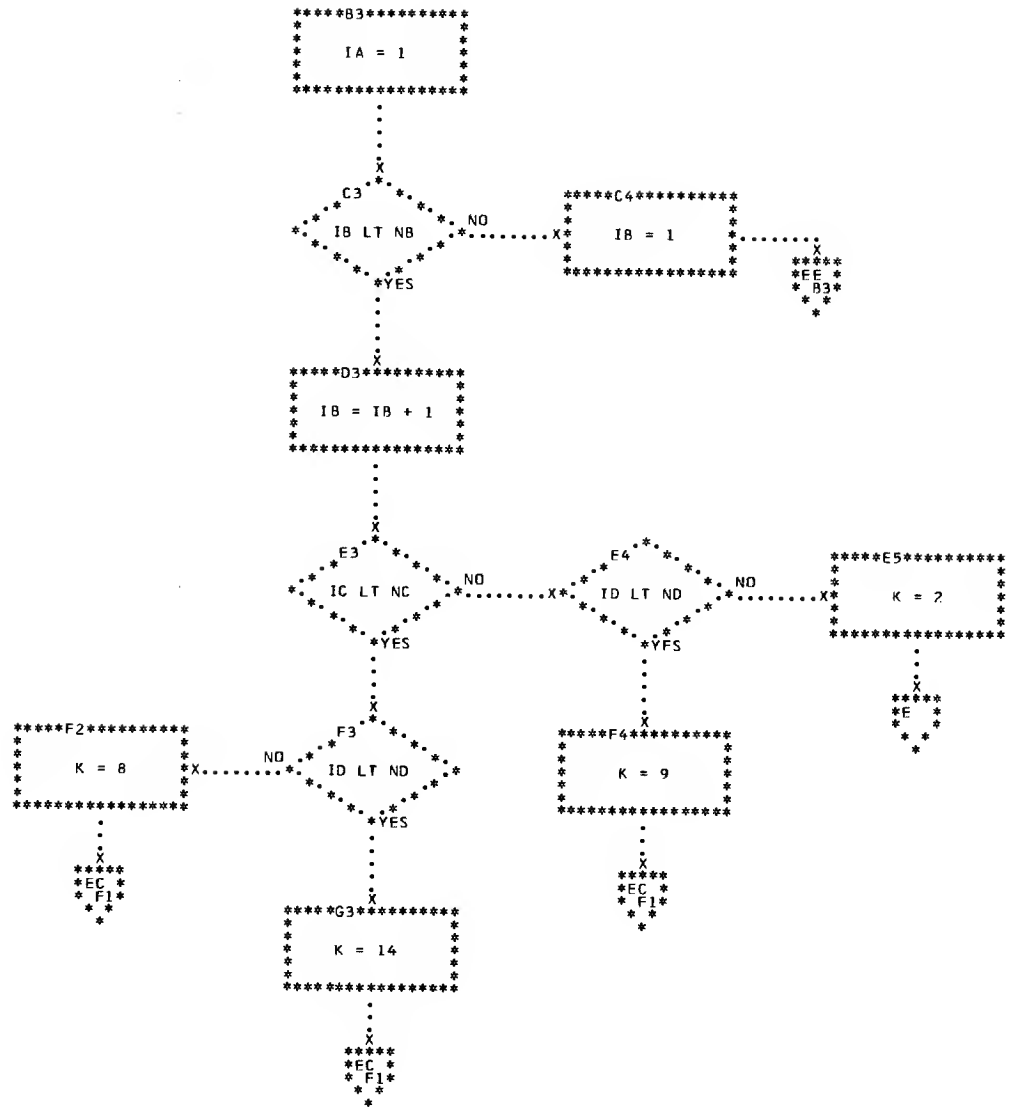


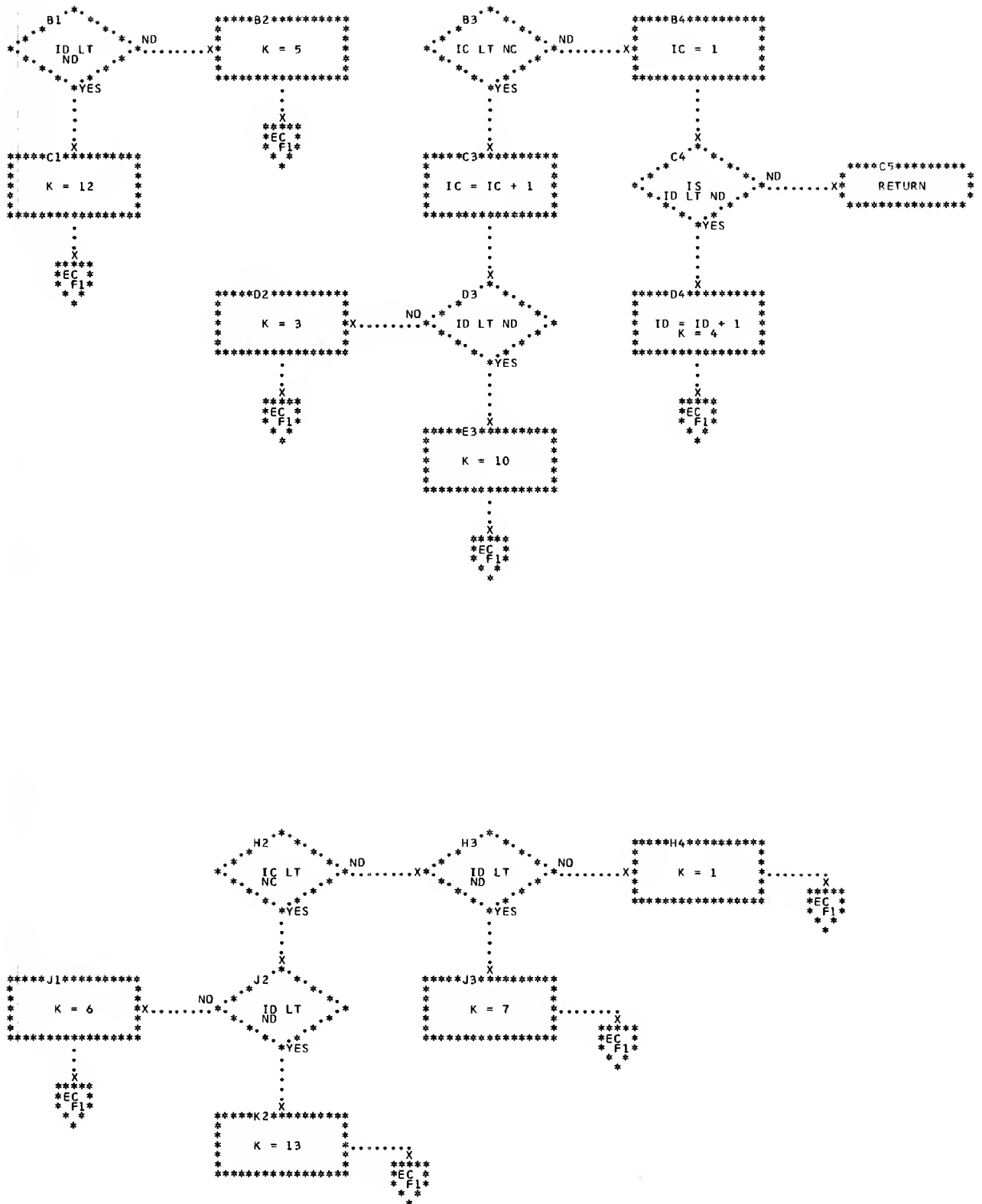








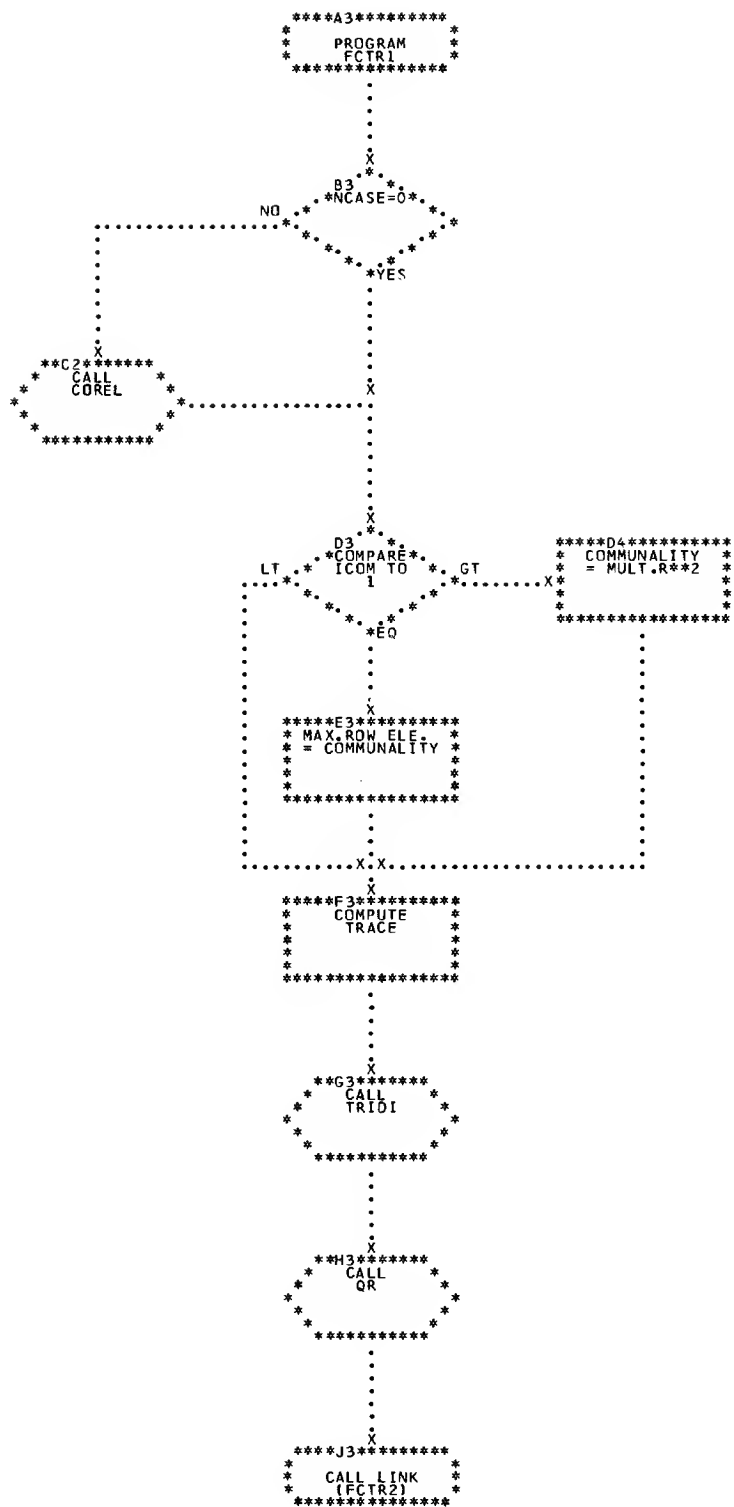


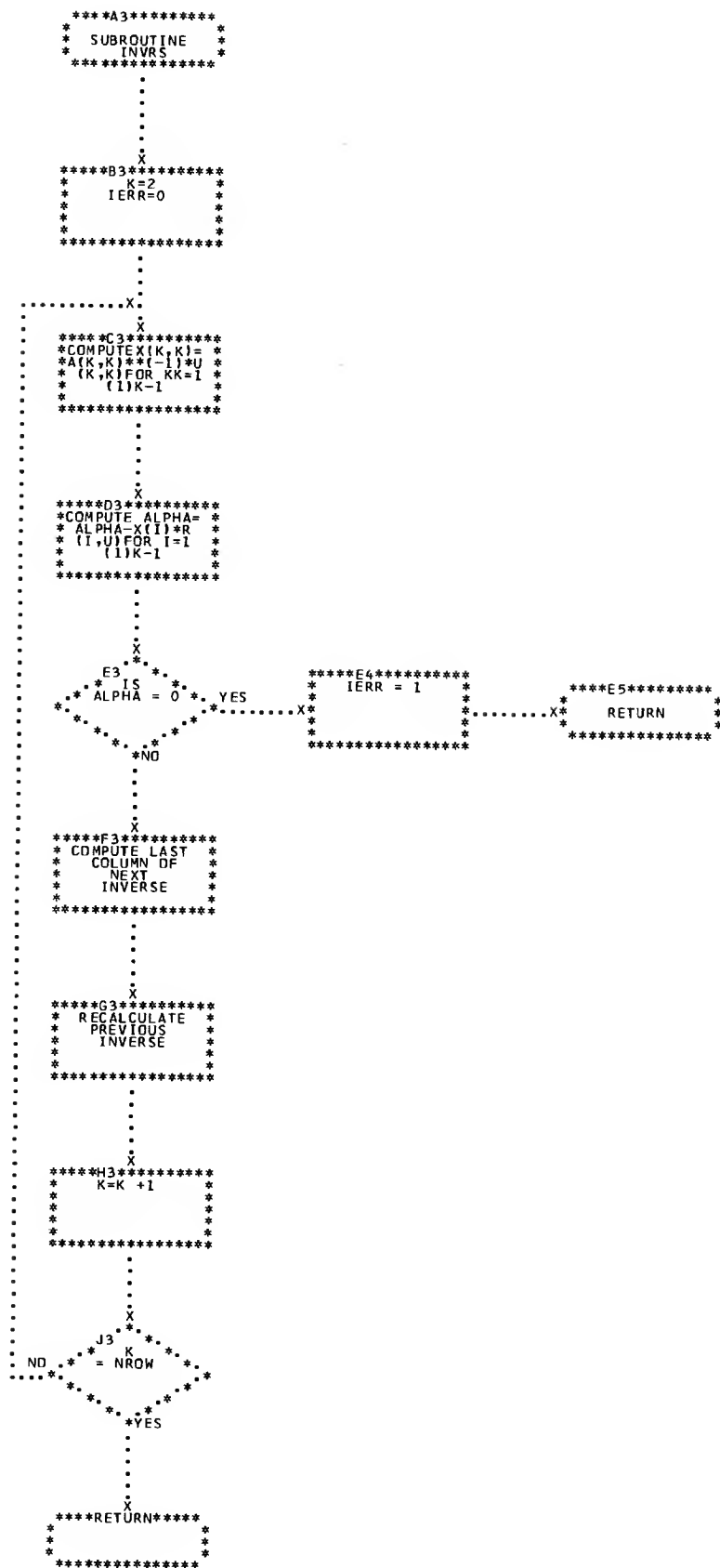


```

*****A2*****
* SUBROUTINE REPT *
*****
      .
      .
      X
*****B2*****
* GENERATE DEGS OF FREEDOM VECTOR *
*****
      .
      .
      X
*****C2*****
* COMPUTE SUM OF SQUARES DIVISOR *
*****
      .
      .
      X
*****D2*****
* INITIALIZE COUNTERS *
*****
      .
      .
      X
*****E2*****
* COMPUTE TOTAL SUM OF SQUARES *
*****
      .
      .
      X
*****F2*****
* READ COMPONENT LINE CARD *
*****
      .
      .
      X
*****G2*****
* SMSQ = 0 *
* NOFI = 0 *
* S4SQM = 0 *
*****
      .
      .
      X
*****H2*****
* ADD COMPONENTS TO SMSQ, *
* NOFI, S4SQM, FROM LINE CARD *
*****
      .
      .
      X
*****I2*****
* PRINT TITLE AND COLUMN HEAD *
*****
      .
      .
      X
*****J2*****
* PRINT LINE COMPONENT *
*****
      .
      .
      X
*****K2*****
* ACCUM TOTAL AND DEGREES OF FREEDOM *
*****
      .
      .
      X
*****L2*****
* F4 IS INDI LT O *
      YES
      .
      .
      X
*****M2*****
* G4 NEED RESIDUAL LINE *
      YES
      .
      .
      X
*****N2*****
* H4 PRINT RESIDUAL *
*****
      .
      .
      X
*****O2*****
* J4 PRINT TOTAL SUM OF SQUARES *
*****
      .
      .
      X
*****P2*****
* K4 RETURN *
*****

```





```

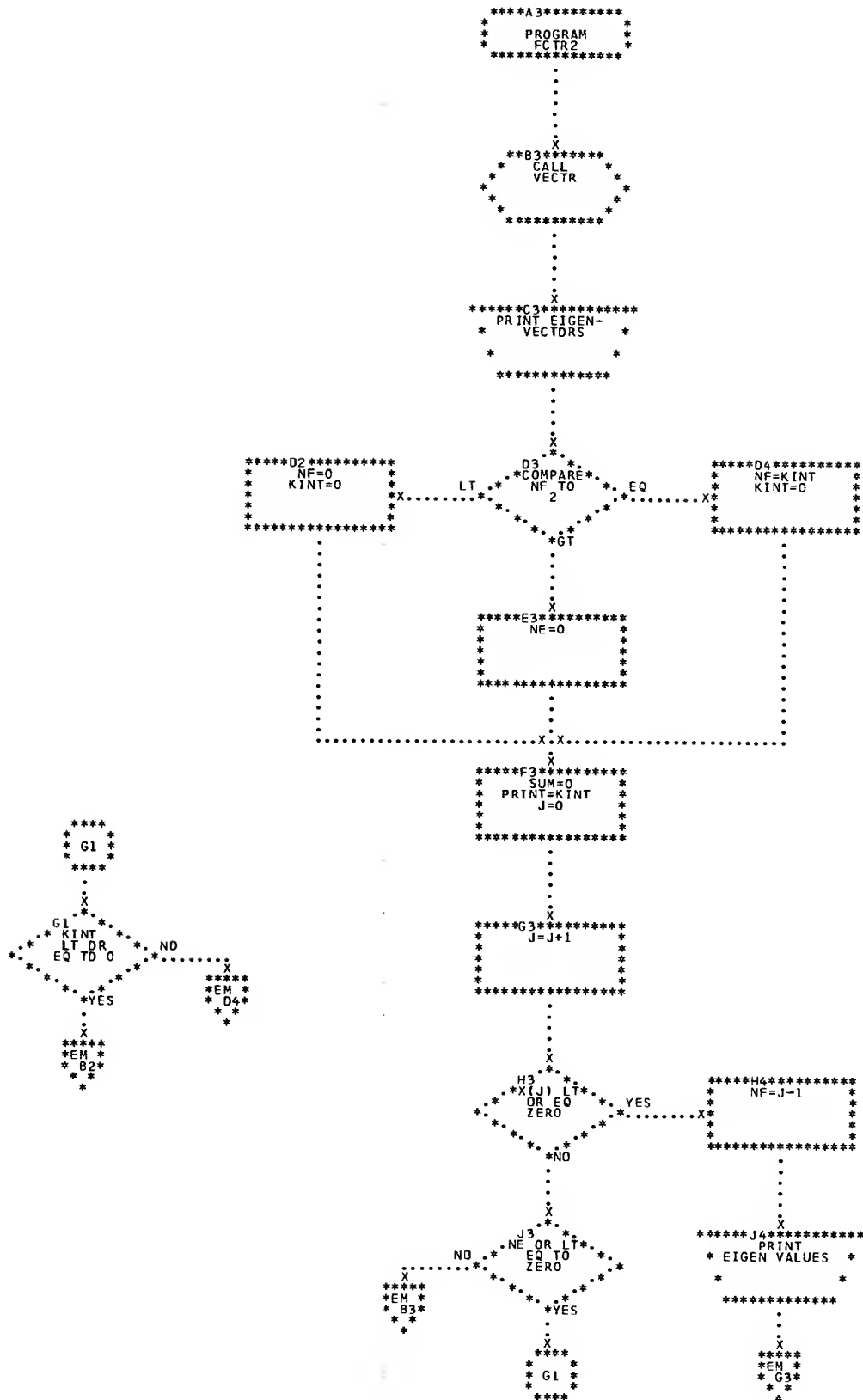
*****A2*****
* SUBROUTINE *
* TRIOI *
*****
.
.
.
.
.
X
*****B2*****
* COMPUTE *
* N-2 ELEM. *
* ORTH. TRANS. *
* *
*****
.
.
.
.
.
X
*****C2*****
* APPLY THESE *
* TO AN *
* IDENTITY *
* MATRIX *
*****
.
.
.
.
.
X
*****O2*****
* RETURN *
* *
*****

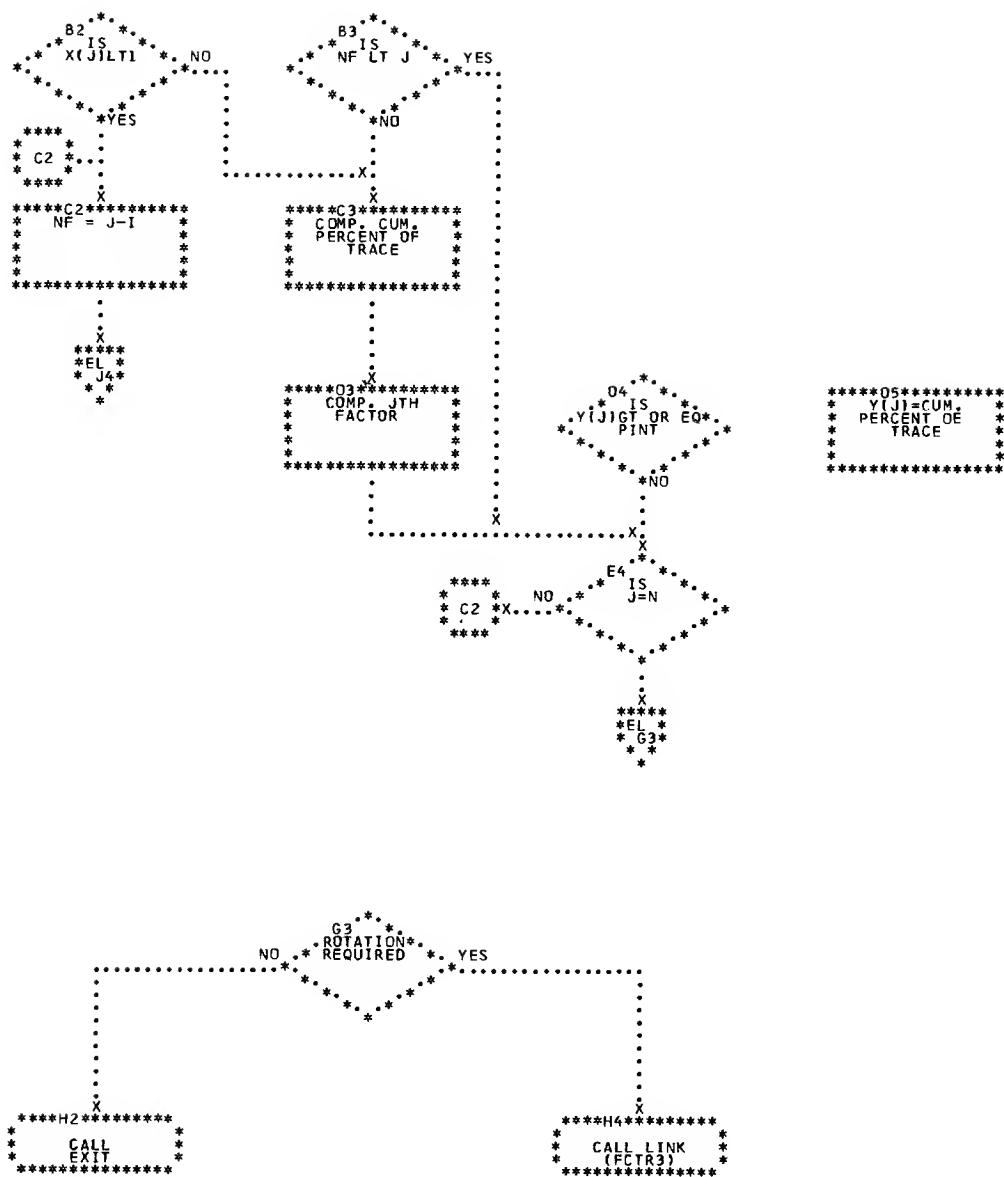
```

```

*****A4*****
* SUBROUTINE *
* OR *
*****
.
.
.
.
.
X
*****B4*****
* SET INTERNAL *
* ARRAYS *
* FROM TRIOI *
* *
*****
.
.
.
.
.
X
*****C4*****
* HANDLE 2X2 *
* BLOCKS *
* SEPARATELY *
* *
*****
.
.
.
.
.
X
*****O4*****
* APPLY SHORTCUT *
* SINGLE OR *
* ITERATION *
* *
*****
.
.
.
.
.
X
*****E4*****
* ORDER *
* THE *
* EIGEN VALUES *
* *
*****
.
.
.
.
.
X
*****F4*****
* RETURN *
* *
*****

```





```

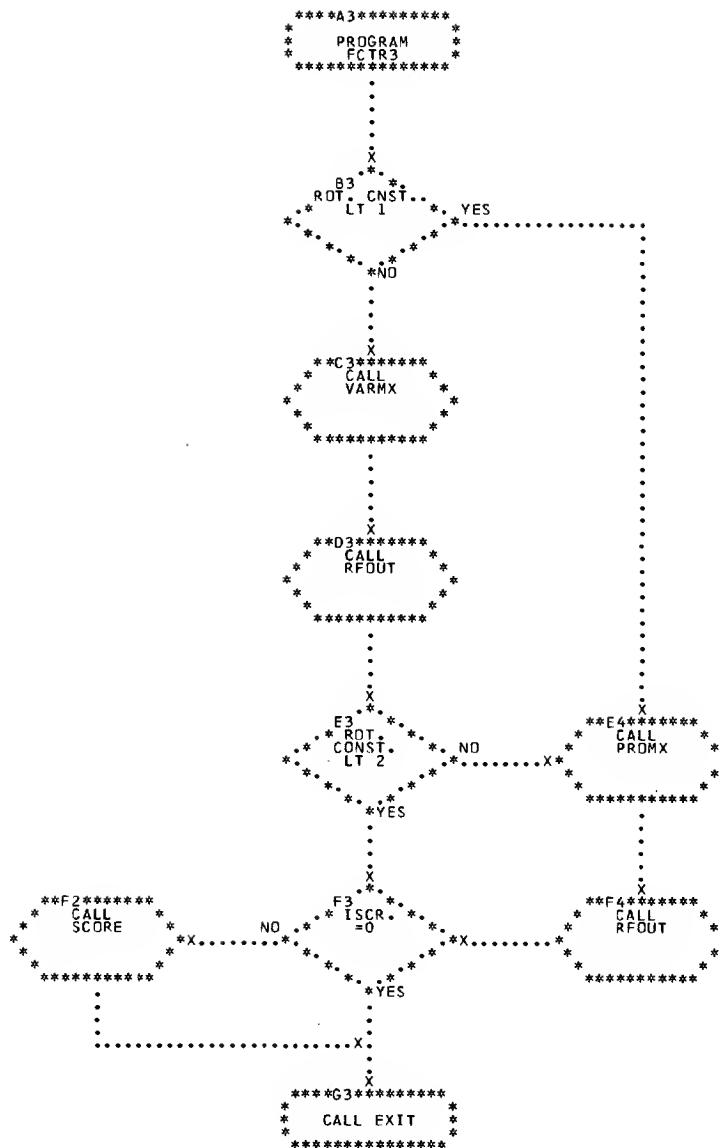
*****A1*****
* SUBROUTINE *
* VECTR *
*****
      .
      .
      .
      .
      X
*****B1*****
* INITIARS *
* OF EVAT. *
* TO ONES *
*****
      .
      .
      .
      .
      X
*****C1*****
* GET *
* APPROX. *
* SOLUTION *
*****
      .
      .
      .
      .
      X
*****D1*****
* REFINE *
* SOLUTION *
*****
      .
      .
      .
      .
      X
*****E1*****
* NORMALIZE *
* EIGENVECTOR *
*****
      .
      .
      .
      .
      X
*****F1*****
* RETURN *
*****

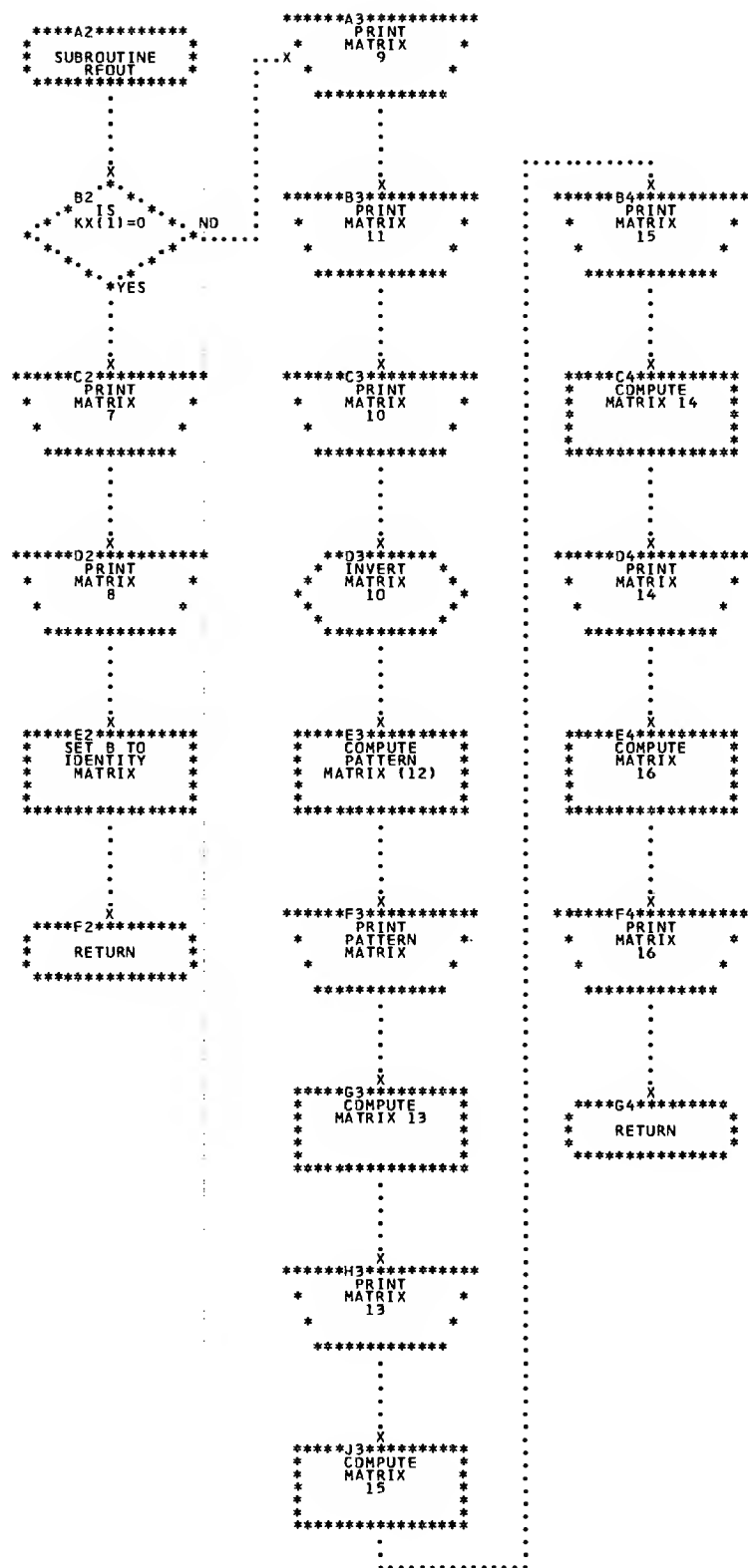
```

```

*****A3*****
* SUBROUTINE COVE *
*****
      .
      .
      .
      .
      X
*****B3*****
* INITIALIZE *
*****
*           *
*           *
*           *
*****
*****C3*****
* SOLVE *
* SIM. TRIODI EQUATIONS *
*****
      .
      .
      .
      .
      X
*****D3*****
* RETURN *
*****

```





```

*****A2*****
*      SUBROUTINE      *
*      PROMX           *
*****

```

```

      .
      .
      .
      .
      X
*****B2*****
**   B=A**(TH)   **
**               **
**               **
**               **
*****

```

```

      .
      .
      .
      .
      X
*****G2*****
      8*(-1)
*****
*
*
*
*
*****

```

```

      .
      .
      .
      .
      X
*****02*****
*      COMPUTE      *
*      ROW NORM-    *
*      ALIZING      *
*      VECTOR,H     *
*****

```

```

      .
      .
      .
      .
      X
*****E2*****
*      COLUMN      *
*    NORMALIZING    *
*      VECTOR,G     *
*                    *
*****

```

```

      .
      .
      .
      .
      X
*****F2*****
*      NORMALIZE      *
*    ROWS, COLS.    *
*      OF A          *
*                    *
*****

```

```

      .
      .
      .
      X
*****G2*****
E=A**{T+K}
*****

```

```

      .
      .
      .
      .
      X
*****H2*****
**   TRANSFORM-   **
**     ATION      **
**    MATRIX      **
**   B=B*E        **
*****

```

```

      .
      .
      .
      .
      X
*****J2*****
**   NORMALIZE   **
** COLS OF       **
**     B         **
*****

```

```

*****
      X
*****B4*****
*      ROTATE TO      *
*    REF. VCTR.      *
*    STRUCTURE        *
*      MATRIX         *
*****
*****

```

```

      .
      .
      .
      .
      X
*****C4*****
*      COMPUTE      *
*      CORREL      *
*      E=B**(T+1)   *
*                  *
*                  *
*****

```

```

      .
      .
      .
      .
      X
****D4*****
*          *
*  RETURN  *
*          *
*****

```

```

*****G3*****
*      K IS      *
*      OBLIQUE-  *
X*      NESS     *
*      POWER     *
*               *
*****

```

```

*****A3*****
*               *
*  SUBROUTINE   *
*    VARMX     *
*               *
*****

```

```

X
*****B3*****
*      INITIALIZE      *
*      INTERNAL CON-   *
*      STANTS AND      *
*      VARIABLES       *
*                      *
*****

```

```

*****C3*****
*  INITIALIZE  *
*  T MATRIX   *
*  TO IDENTITY *
*             *
*****

```

```

X
*****03*****
*      NORMALIZE      *
*      INPUT          *
*      MATRIX A       *
*                      *
*****

```

```

X
*****E3*****
*  COMPUTE VAR-  *
*  IANCE FOR    *
*  EACH COLUMN  *
*              *
*              *
*****

```

```

*****F3*****
*  COMPUTE SUM  *
*  OF VARIANCES  *
*    OVER ALL   *
*   COLUMNS    *
*****

```

G3 HAVE  
50 CYCLES  
ELAPSED

```

*****G4*****
*      KX(1) = 0      *
X*                    *
*                    *
*                    *
*                    *
*****

```

```

*****G5*****
*               *
*      RETURN      *
*               *
*****

```

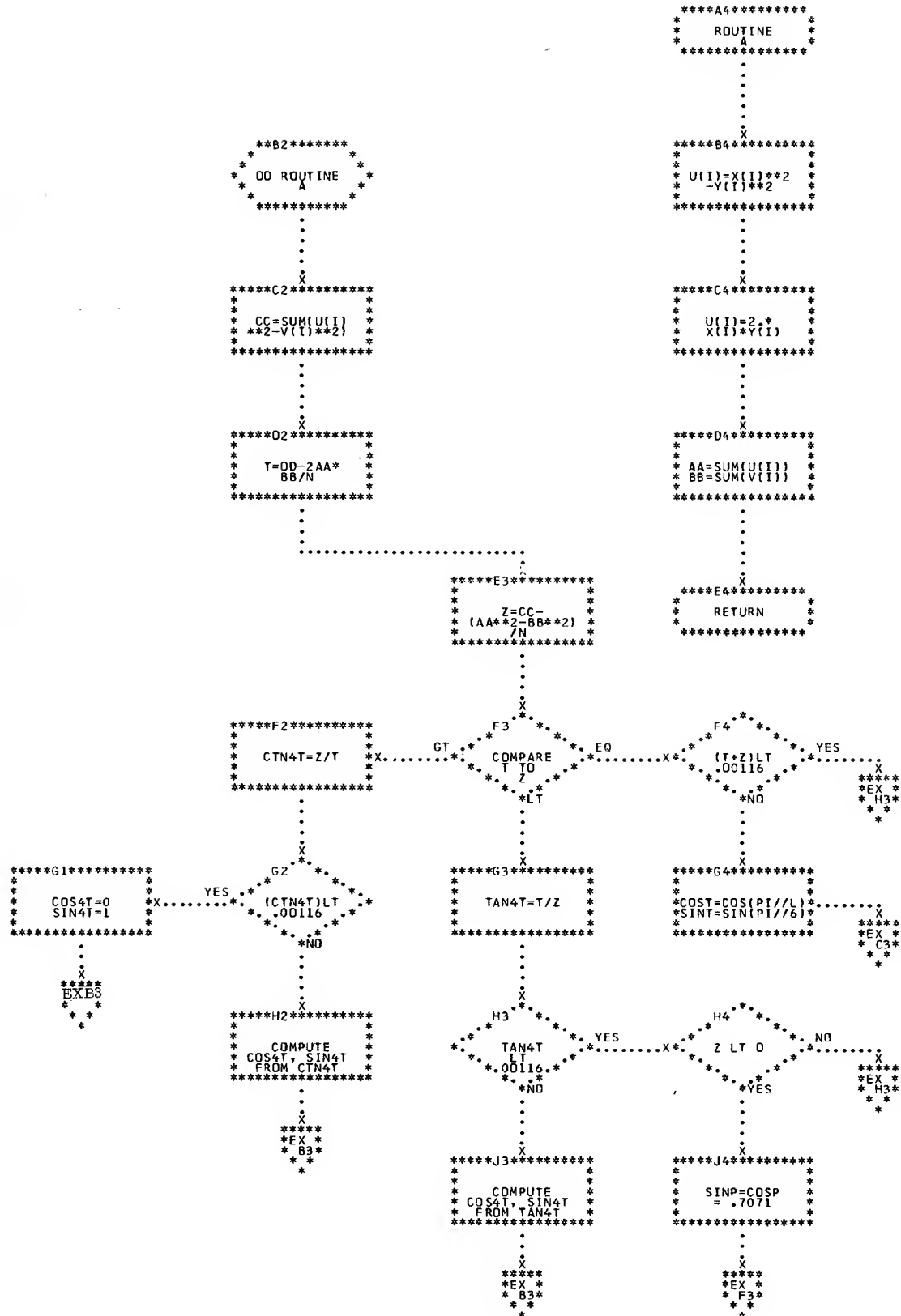
H3 IS  
SUM=SUM Y  
ON PREVIOUS  
CYCLE  
NO

```

*****J3*****
PICK 2 COLS.
FROM A FOR
ROTATION
(X,Y)
*****

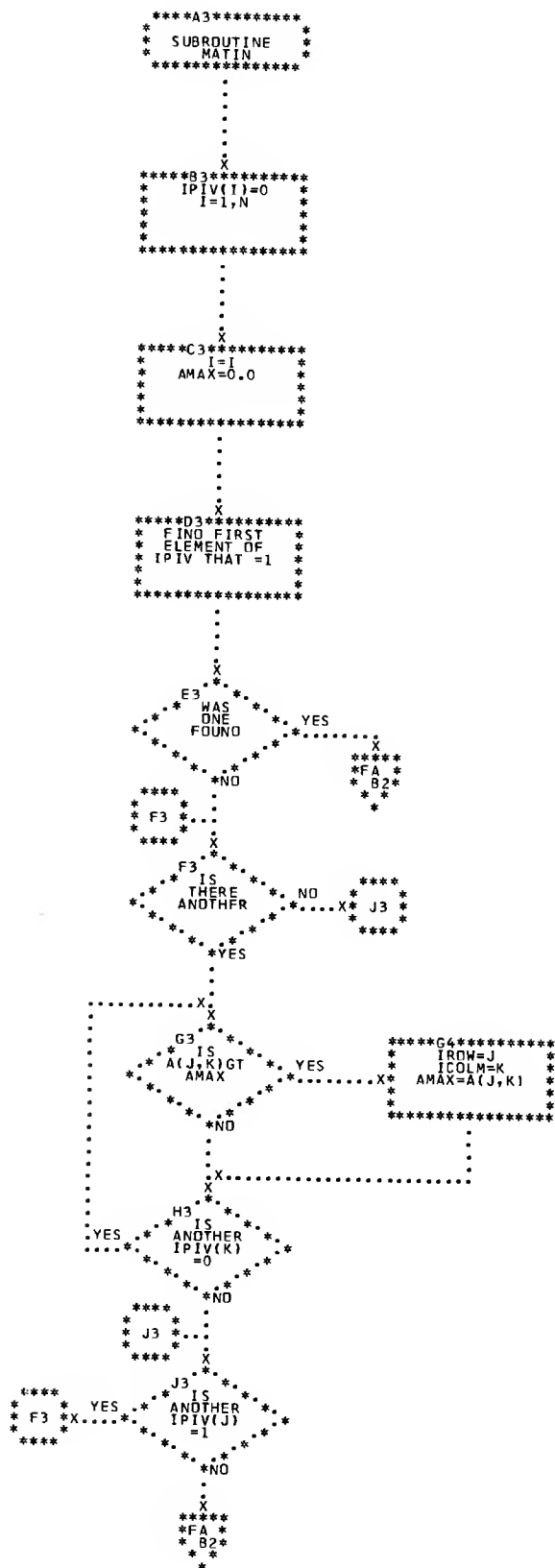
```

X  
\*\*\*\*\*  
\*EW\*  
\*BZ\*  
\*  
\*

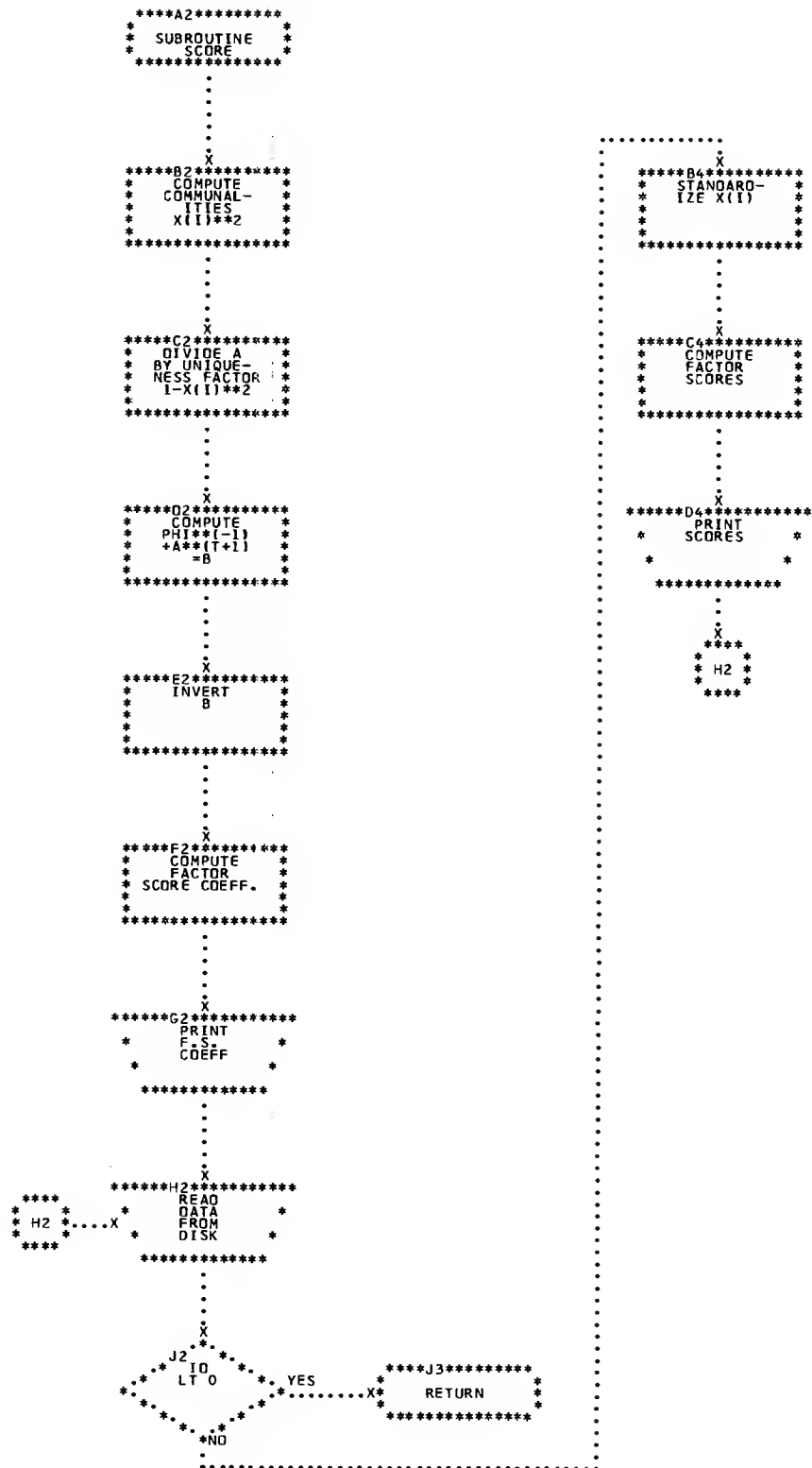














**IBM**

International Business Machines Corporation  
Data Processing Division  
112 East Post Road, White Plains, N.Y. 10601  
(USA Only)

IBM World Trade Corporation  
821 United Nations Plaza, New York, New York 10017  
(International)